ECE 683-001 Fall 2005 Computer Network Design and Analysis

Lecture 2: Applications and Layered Architectures

Professor Nirwan Ansari
Department of Electrical and Computer Engineering
New Jersey Institute of Technology

Sources:

- **Communication Networks 2nd Edition Chapter 2 by Alberto Leon-Garcia and Indra Widjaja**

Protocols, Services & Layering
OSI Reference Model
TCP/IP Architecture
How the Layers Work Together
Berkeley Sockets
Application Layer Protocols & Utilities

Lecture 2 Applications and Layered Architectures

Protocols, Services & Layering

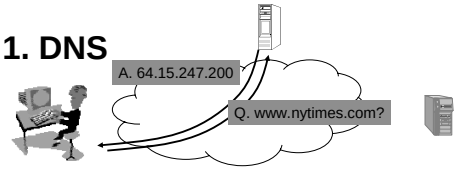
Layers, Services & Protocols

- The overall communications process between two or more machines connected across one or more networks is very complex
- ***Layering*** partitions related communications functions into groups that are manageable
- Each layer provides a ***service*** to the layer above
- Each layer operates according to a ***protocol***
- Let's use examples to show what we mean

Web Browsing Application

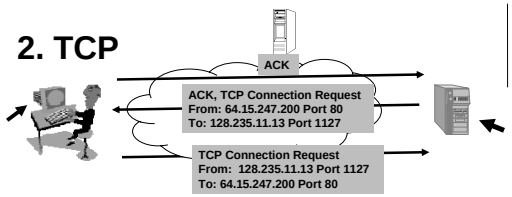
- World Wide Web allows users to access resources (i.e. documents) located in computers connected to the Internet
- Documents are prepared using HyperText Markup Language (HTML)
- A browser application program is used to access the web
- The browser displays HTML documents that include *links* to other documents
- Each link references a *Uniform Resource Locator* (URL) that gives the name of the machine and the location of the given document
- Let's see what happens when a user clicks on a link

1. DNS



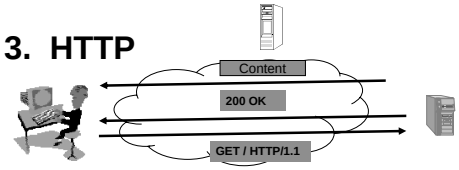
- User clicks on http://www.nytimes.com/
- URL contains Internet name of machine (www.nytimes.com), but not Internet address
- Internet needs Internet address to send information to a machine
- Browser software uses Domain Name System (DNS) protocol to send query for Internet address
- DNS system responds with Internet address

2. TCP



- Browser software uses HyperText Transfer Protocol (HTTP) to send request for document
- HTTP server waits for requests by listening to a well-known port number (80 for HTTP)
- HTTP client sends request messages through an "ephemeral port number," e.g. 1127
- HTTP needs a Transmission Control Protocol (TCP) connection between the HTTP client and the HTTP server to transfer messages reliably

3. HTTP



- HTTP client sends its request message: "GET …"
- HTTP server sends a status response: "200 OK"
- HTTP server sends requested file
- Browser displays document

- Clicking a link sets off a chain of events across the Internet!
- Let's see how protocols & layers come into play…

Protocols

- A *protocol* is a set of rules that governs how two or more communicating entities in a layer are to interact
- *Messages* that can be sent and received
- *Actions* that are to be taken when a certain event occurs, e.g. sending or receiving messages, expiry of timers
- **The purpose of a protocol is to provide a service to the layer above**

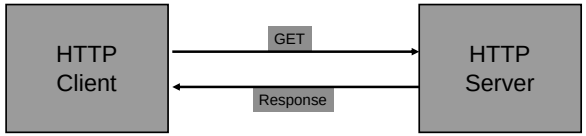
Layers

- A set of related communication functions that can be managed and grouped together
- Application Layer: communications functions that are used by application programs
  - HTTP, DNS, SMTP (email)
- Transport Layer: end-to-end communications between two processes in two machines
  - TCP, User Datagram Protocol (UDP)
- Network Layer: node-to-node communications between two machines
  - Internet Protocol (IP)

Example: HTTP

- HTTP is an application layer protocol
- Retrieves documents on behalf of a browser application program
- HTTP specifies fields in request messages and response messages
  - Request types; Response codes
  - Content type, options, cookies, …
- HTTP specifies actions to be taken upon receipt of certain messages

HTTP Protocol



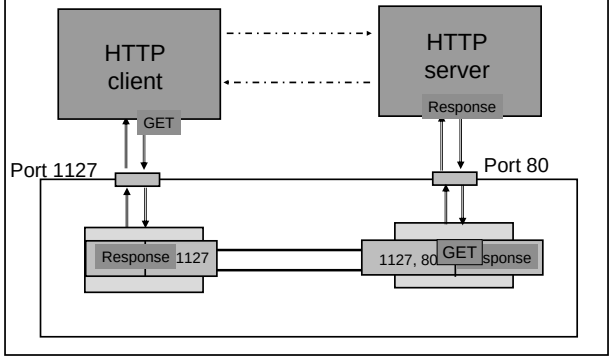
- HTTP assumes messages can be exchanged directly between HTTP client and HTTP server
- In fact, HTTP client and server are processes running in two different machines across the Internet
- HTTP uses the reliable stream transfer service provided by TCP

Example: TCP

- TCP is a transport layer protocol
- Provides *reliable byte stream service* between two processes in two computers across the Internet
- Sequence numbers keep track of the bytes that have been transmitted and received
- Error detection and retransmission used to recover from transmission errors and losses
- TCP is *connection-oriented*: the sender and receiver must first establish an association and set initial sequence numbers before data is transferred
- Connection ID is specified uniquely by

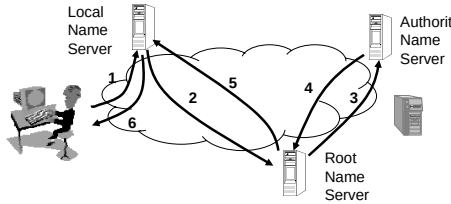
(send port #, send IP address, receive port #, receiver IP address)

HTTP uses service of TCP



Example: DNS Protocol

- DNS protocol is an application layer protocol
- DNS is a distributed database that resides in multiple machines in the Internet
- DNS protocol allows queries of different types
  - Name-to-address or Address-to-name
  - Mail exchange
- DNS usually involves short messages and so uses service provided by UDP
- Well-known port 53



- Local Name Server: resolve frequently-used names
  - University department, ISP
  - Contacts Root Name server if it cannot resolve query
- Root Name Servers: 13 globally
  - Resolves query or refers query to Authoritative Name Server
- Authoritative Name Server: last resort
  - Every machine must register its address with at least two authoritative name servers

Example: UDP

- UDP is a transport layer protocol
- Provides *best-effort datagram service* between two processes in two computers across the Internet
- Port numbers distinguish various processes in the same machine
- UDP is *connectionless*
- Datagram is sent immediately
- Quick, simple, but not reliable

Summary

- Layers: related communications functions
  - Application Layer: HTTP, DNS
  - Transport Layer: TCP, UDP
  - Network Layer: IP
- Services: a protocol provides a communications service to the layer above
  - TCP provides connection-oriented reliable byte transfer service
  - UDP provides best-effort datagram service
- Each layer builds on services of lower layers
  - HTTP builds on top of TCP
  - DNS builds on top of UDP
  - TCP and UDP build on top of IP

Lecture 2
Applications and Layered Architectures

OSI Reference Model

Why Layering?

- Layering simplifies design, implementation, and testing by partitioning overall communications process into parts
- Protocol in each layer can be designed separately from those in other layers
- Protocol makes “calls” for services from layer below
- Layering provides flexibility for modifying and evolving protocols and services without having to change layers below
- Monolithic non-layered architectures are costly, inflexible, and soon obsolete

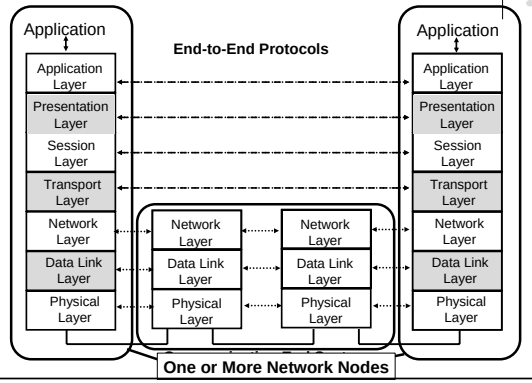
Open Systems Interconnection

- Network architecture:
  - Definition of all the layers
  - Design of protocols for every layer
- By the 1970s every computer vendor had developed its own proprietary layered network architecture
- Problem: computers from different vendors could not be networked together
- Open Systems Interconnection (OSI) was an international effort by the International Organization for Standardization (ISO) to enable multivendor computer interconnection

OSI Reference Model

- Describes a seven-layer abstract reference model for a network architecture
- Purpose of the reference model was to provide a framework for the development of protocols
- OSI also provided a unified view of layers, protocols, and services which is still in use in the development of new protocols
- Detailed standards were developed for each layer, but most of these are not in use
- TCP/IP protocols preempted deployment of OSI protocols

7-Layer OSI Reference Model

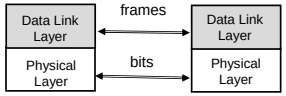


Physical Layer

- Transfers bits across link
- Definition & specification of the physical aspects of a communications link
  - Mechanical: cable, plugs, pins...
  - Electrical/optical: modulation, signal strength, voltage levels, bit times, …
  - functional/procedural: how to activate, maintain, and deactivate physical links…
- Ethernet, DSL, cable modem, telephone modems…
- Twisted-pair cable, coaxial cable optical fiber, radio, infrared, …

Data Link Layer

- Transfers *frames* across *direct* connections
- Groups bits into frames
- Detection of bit errors; Retransmission of frames
- Activation, maintenance, & deactivation of data link connections
- Medium access control for local area networks
- Flow control

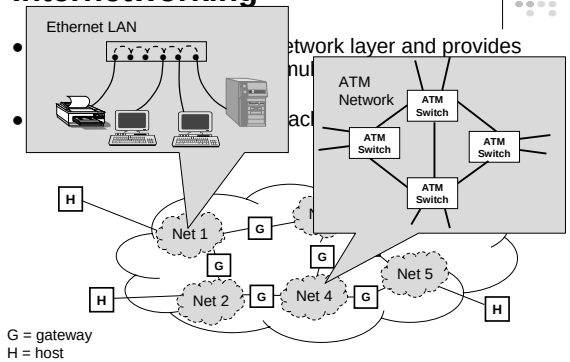


Network Layer

- Transfers *packets* across multiple links and/or multiple networks
- Addressing must scale to large networks
- Nodes *jointly* execute routing algorithm to determine paths across the network
- Forwarding transfers packet across a node
- Congestion control to deal with traffic surges
- Connection setup, maintenance, and teardown when connection-based

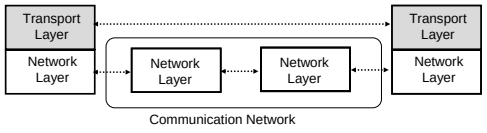
Internetworking

- etwork layer and provides
- ac



Transport Layer

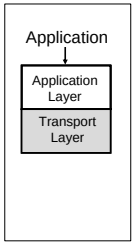
- Transfers data end-to-end from process in a machine to process in another machine
- Reliable stream transfer or quick-and-simple single-block transfer
- Port numbers enable multiplexing
- Message segmentation and reassembly
- Connection setup, maintenance, and release



Application & Upper Layers

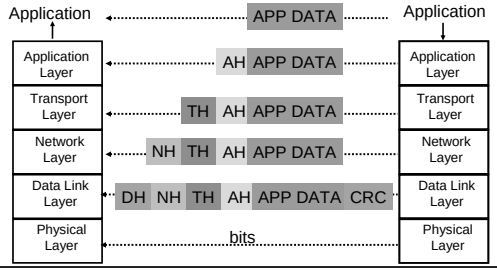
- Application Layer: Provides services that are frequently required by applications: DNS, web acess, file transfer, email…
- Presentation Layer: machine-independent representation of data…
- Session Layer: dialog management, recovery from errors, …

Incorporated into Application Layer



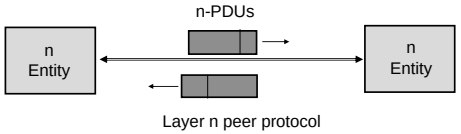
Headers & Trailers

- Each protocol uses a header that carries addresses, sequence numbers, flag bits, length indicators, etc…
- CRC check bits may be appended for error detection



OSI Unified View: Protocols

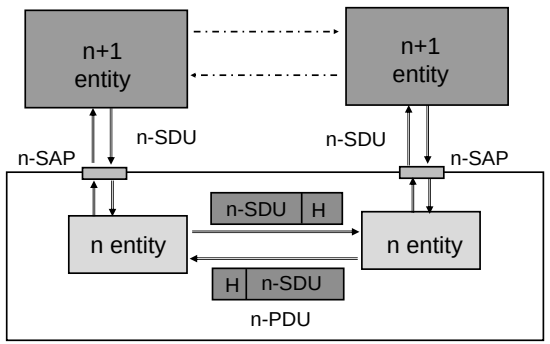
- Layer n in one machine interacts with layer n in another machine to provide a service to layer n +1
- The entities comprising the corresponding layers on different machines are called *peer processes.*
- The machines use a set of rules and conventions called the *layer-n protocol*.
- Layer-n peer processes communicate by exchanging *Protocol Data Units* (PDUs)



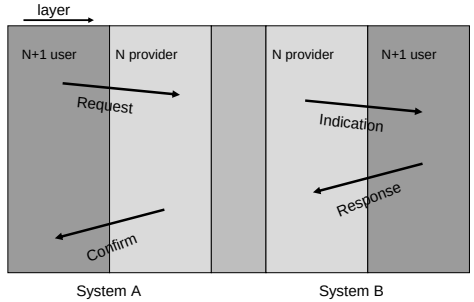
OSI Unified View: Services

- Communication between peer processes is virtual and actually indirect
- Layer n+1 transfers information by invoking the services provided by layer n
- Services are available at *Service Access Points (*SAP's)
- Each layer passes data & control information to the layer below it until the physical layer is reached and transfer occurs
- The data passed to the layer below is called a *Service Data Unit* (SDU)
- SDU's are *encapsulated* in PDU's

Layers, Services & Protocols



Interlayer Interaction

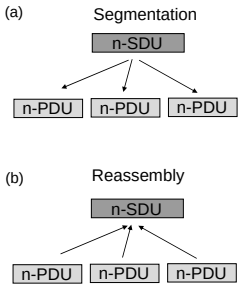


Connectionless & Connection-Oriented Services

- Connection-Oriented
  - Three-phases:
    1. Connection setup between two SAPs to initialize state information
    2. SDU transfer
    3. Connection release
  - E.g. TCP, ATM
- Connectionless
  - Immediate SDU transfer
  - No connection setup
  - E.g. UDP, IP
- Layered services need not be of same type
  - TCP operates over IP
  - IP operates over ATM

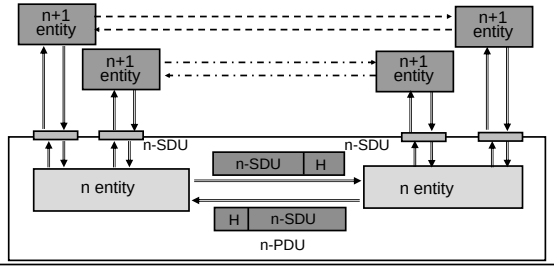
Segmentation & Reassembly

- A layer may impose a limit on the size of a data block that it can transfer for implementation or other reasons
- Thus a layer-n SDU may be too large to be handled as a single unit by layer-(n-1)
- Sender side: SDU is segmented into multiple PDUs
- Receiver side: SDU is reassembled from sequence of PDUs



Multiplexing

- Sharing of layer n service by *multiple* layer n+1 users
- Multiplexing tag or ID required in each PDU to determine which users an SDU belongs to



Summary

- Layers: related communications functions
  - Application Layer: HTTP, DNS
  - Transport Layer: TCP, UDP
  - Network Layer: IP
- Services: a protocol provides a communications service to the layer above
  - TCP provides connection-oriented reliable byte transfer service
  - UDP provides best-effort datagram service
- Each layer builds on services of lower layers
  - HTTP builds on top of TCP
  - DNS builds on top of UDP
  - TCP and UDP build on top of IP