Introduction to Network Programming

CST 357/457 – Systems Programming

Michael Ruth, Ph.D.

Associate Professor

Computer Science & I.T.

mruth@roosevelt.edu

Objectives

- Discuss basic terminology, communication paradigms and protocol layering
- Explain the important elements of the network layer including addressing and the transport layer including protocols and port numbers
- Discuss and compare application protocols and introduce the standardization of protocols
- Introduce network programming architectural models including client/server and peer to peer
- Discuss general networking applications

Terminology

- Two or more computer hardware resources are connected form a **computer network**
- Every machine on a network is a ***node***
  - Nodes which are computers are called ***hosts***
- Every node/host has an ***address***
  - ***Uniquely*** identifies it to the rest of the network
- Some addresses have ***names***
  - identifies an address to make it easier for humans to use

More Terminology

- Most modern networks are ***packet-switched*** networks
  - All data is broken into packets
  - Packets are managed separately
    - Packets contain addresses (to/from)
- ***Protocol***: A precise set of rules governing how two computers communicate:
  - Format of addresses/messages
  - Order in which they are exchanged

Two Basic Communication Paradigms

- Connection-oriented
  - Paradigm
    - Form a ‘‘connection’’ through the network
    - Send / receive data over the connection
    - Terminate the connection
  - Can guarantee bandwidth
- Connectionless
  - Paradigm
    - Form ‘‘packet’’ of data
    - Pass to network
  - Each packet travels independently
  - Packet includes identification of the destination
  - Each packet can be a different size
    - The maximum packet size is fixed (some technologies limit packet sizes to 1,500 octets or less)

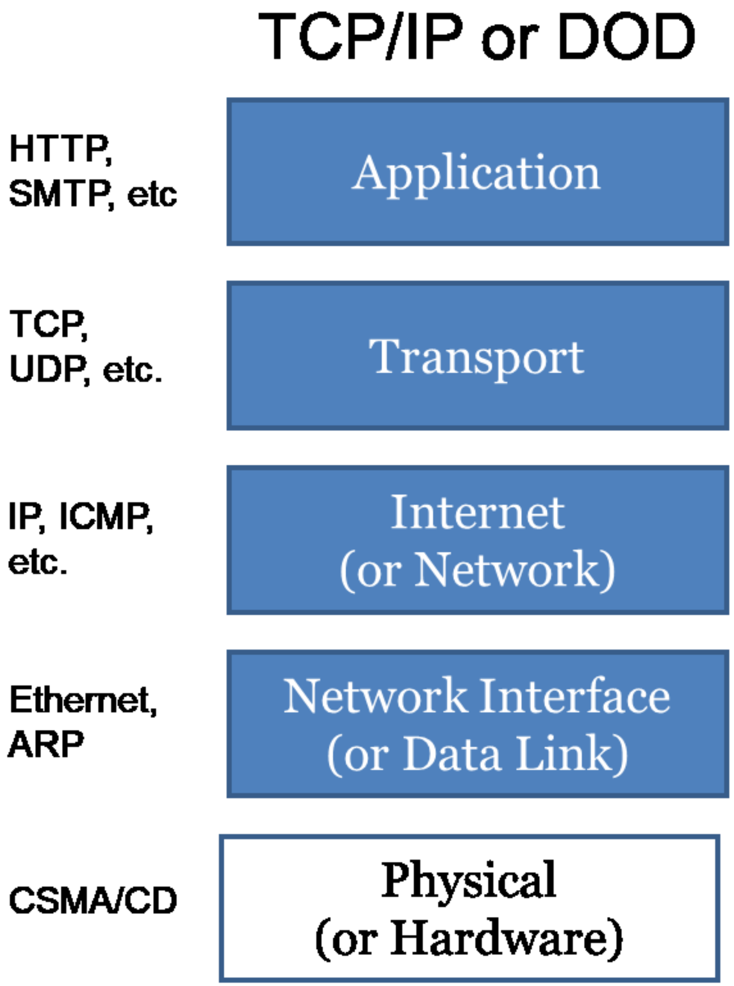
Why so many protocols?

- Communication is difficult to understand
- Many sub-problems
  - Hardware failure
  - Network congestion
  - Packet delay or loss
  - Data corruption
  - Data duplication or inverted arrivals

Why so many protocols? (cont.)

- Divide & Conquer (sort of)
  - Divide the problem into pieces
  - Solve sub-problems separately
  - Combine into integrated whole
- Result is ***layered protocols***
  - Separates protocol functionality
  - Each layer solves one part of the communication problem
  - Intended primarily for protocol designers
  - Set of layers is called a protocol stack

TCP/IP Model



TCP/IP or DOD

HTTP, SMTP, etc — Application

TCP, UDP, etc. — Transport

IP, ICMP, etc. — Internet (or Network)

Ethernet, ARP — Network Interface (or Data Link)

CSMA/CD — Physical (or Hardware)

Network Layer

- Internet Protocol
  - Connection-less
  - “Best Effort” Delivery
    - No guarantee on order, delivery, duplication
  - Provides machine to machine communication
    - Routing
      - What path is followed by packets from source to destination
    - Congestion
      - Controls the number packets in each network

- Three basic components
  - ***Naming (addressing)***
  - Data structure (packet structure)
  - Algorithm (how that packet moves through system)

Routing requires a Destination…

- Each system on a network must be addressable for packets to be delivered
  - IP Addresses provide this form of identification for all hosts in a TCP/IP system
    - 32-bit binary value
      - Values chosen to make routing efficient
    - Address is always divided into two parts
      - Prefix (network ID) identifies network to which host attaches
      - Suffix (host ID) identifies host on that network
    - Generally represents each octet in decimal separated by periods (dots)
      - EX: 192.168.0.23

Division into Suffix/Prefix

- Dividing an address into suffix/prefix is not as easy as it seems
  - – Original scheme used notion of classes
    - Requirement for netid part of address to be exactly 1,2,3 bytes long is very problematic
      - – Leads to
        - » poor utilization of assigned address space
        - » rapid depletion of address space
  - – New scheme uses CIDR addressing and subnetting concepts to allow for:
    - Netid can be any number of bits long
      - – Rather than simply 8, 16, or 24

CIDR Block/Subnet Mask

- If the division is then arbitrary, how do we determine where the division occurs?
  - IE which part is the netid and which part is the host id?

- An address mask is used
  - Store address mask with each route
    - AKA subnet mask
  - Send pair of (address, mask) whenever exchanging routing information
  - Known as a CIDR block

CIDR Notation

- Addresses written NUMBER/m
  - NUMBER is IP prefix
  - m is ‘‘address mask’’ length

  - For example:
    - IP Address: 192.60.128.0
    - Subnet Mask: 255.255.252.0

    - Now: 192.60.128.0/22

Addressing for Humans

- Symbolic names are easier to remember
  - Designed to remain the same even if the numeric address changes
  - Must be unique for each host on the Internet

- Requires an infrastructure to translate hostnames into IP addresses…

Domain Name System (DNS)

- Required to translate symbolic names to equivalent IP addresses
  - DNS implements a distributed database of name-to-address mappings for lookups
  - DNS also refers to the infrastructure used to support the distributed database of IP address to host name mappings

Transport Layer

- Provides end-to-end connection from application program to application program
  - Often handles reliability, flow control
  - Protocols are TCP and UDP
  - Differentiate host applications on the same server using ***port numbers***
    - Servers listen on ports
    - Clients connect to those ports to use servers

UDP vs TCP

UDP

- Connection-less
  - “Best Effort”
    - Delivery/Order not guaranteed

TCP

- Connection-oriented
  - notion of “virtual circuit”
  - Guarantees:
    - Delivery & Order

More on Ports!

- Server ports are well-known…
  - IE, port number 80 for HTTP
    - The server will listen on port 80

- However, client ports are
  - Generally use the range 1024 – 65535
    - The OS handles the client request for a port number
  - Temporary!
    - The port is used for the request to a server
      - If the app needs to reconnect, it will request a new port

Application Protocols

- Network applications run on end systems
  - They depend on the network to provide a service
  - … but cannot run software on the network elements
- Network applications run on multiple machines
  - Different end systems communicate with each other
  - Software is often written by multiple parties
- Leading to a need to explicitly define a protocol
  - Types of messages (e.g., requests and responses)
  - Message syntax (e.g., fields, and how to delineate)
  - Semantics of the fields (i.e., meaning of the information)
  - Rules for when and how a process sends messages

Protocols

- Comparing Application Protocols
- Reflecting/reviewing the Application protocols
- Protocol Specification
  - Network/Application Protocol Specification
- Protocol Standardization
  - IETF
  - W3C

Comparing the Protocols

- Commands and replies
  - Telnet sends commands in binary, whereas the other protocols are text based
  - Many of the protocols have similar request methods and response codes

- Data types
  - Telnet, and SMTP transmit text data in standard ASCII
  - SMTP uses MIME standard for sending non-text data
    - HTTP incorporates some key aspects of MIME (e.g., classification of data formats)

Comparing the Protocols (Cont.)

- Transport
  - Telnet, FTP, SMTP, and HTTP all depend on reliable transport protocol
  - Telnet, SMTP, and HTTP use a single TCP connection
- State
  - In Telnet and SMTP, the server retains information about the session with the client
  - In contrast, HTTP servers are stateless

Reflecting on Application Protocols

- Protocols are tailored to the applications
  - Each protocol is customized to a specific need
- Protocols have many key similarities
  - Each new protocol was influenced by the previous ones
  - New protocols commonly borrow from the older ones
- Protocols depend on same underlying substrate
  - Ordered reliable stream of bytes (i.e., TCP)
  - Domain Name System (DNS)
- Relevance of the protocol standards process
  - Important for interoperability across implementations
  - Yet, not necessary if same party writes all of the software
  - …which is increasingly common (e.g., P2P software)

Network Application Design

- Network Applications follow a series of architectural models:
  - **Client/server**
  - Peer to Peer
  - Hybrid Systems

Client/Server Model

- A client initiates a request and the server fulfills the request
  - Imagine going to a bar and ordering a beer

- Basic model
  - Server starts first and waits for contact
  - Clients start second and initiate contact

Characteristics of a Client

- Arbitrary application program
- Becomes client temporarily
- Can also perform other computations
- Invoked directly by user
- Runs locally on user's computer
- Actively initiates contact with a server
- Usually contacts one server at a time

Characteristics of a Server

- Special-purpose, privileged program
- Dedicated to providing one service
- Can handle multiple remote clients simultaneously
- Invoked automatically when system boots
- Executes "forever"
- Waits passively for client contact
- Accepts requests from arbitrary clients
- Needs powerful computer and operating system

Summary

- Discussed basic terminology, communication paradigms and protocol layering
- Explained the important elements of the network layer including addressing and the transport layer including protocols and port numbers
- Discussed and compare application protocols and introduce the standardization of protocols
- Introduced network programming architectural models including client/server and peer to peer
- Discussed general networking applications

Questions?