SE 4C03 Winter 2006

08 Interaction Schemes

Instructor: W. M. Farmer

Revised: 14 March 2007

Client-Server Model

- **Servers** provide services over a network
  - Usually listen at a particular **reserved** TCP and/or UDP port
  - May participate in more than one TCP connection at the same time
  - Servers are often called **daemons** and are then given a name that ends with `d` (e.g., `httpd`)
  - A server may be organized as a group of processes or threads

- **Clients** utilize services provided by servers
  - Clients initiate connections to servers
  - Usually are assigned an **ephemeral** port by the OS

- Servers are usually more complex than clients
  - The burden of security falls mostly on the server

Components of a Client-Server Application

1. Communication protocol
2. Server running a server program
3. One or more clients running client programs
4. Communication channels via TCP or UDP

The Socket Interface

- The **socket interface** is an interface for application programs to establish communication channels using TCP/IP protocols (as well as other protocols)

- The socket interface can be implemented:
  - Directly in the operating system
  - By a set of library routines
  - Within a programming language (e.g., Java)

- A **socket** is the end point of a communication channel
  - Is a generalization of a Unix file

- The socket interface is becoming a de facto standard

Functions for Constructing a Socket

- **Create**: creates a socket
- **Bind**: establishes a local protocol port for a socket
  - Usually only called by a server process
- **Connect**: connects a socket to a destination IP address and protocol port
  - For TCP, a TCP connection is established
  - For UDP, no connection is made, but the destination address and port number are stored
- **Listen**: enables a server process to listen to a socket
- **Accept**: blocks a server process until a connection request arrives and then creates a new socket

Sending and Receiving Data

- Data is sent through a socket using various kinds of **write** functions
- Data is received through a socket using various kinds of **read** functions

Other Socket Interface Functions

- Functions for getting socket attributes such as:
  - Source and destination IP addresses
  - Source and destination IP protocol ports

- Functions for getting and setting socket options such as:
  - Buffer sizes
  - Timeout parameters

- Library functions that provide network services such as:
  - DNS queries
  - Host information
  - Network information
  - Protocol information

- **Close**: closes a socket