SE3910 – REAL TIME SYSTEMS

Networking and Sockets

ROADMAP

- Today
  - An in class demo on sockets / C programming tutorials
    - (http://www.linuxhowtos.org/C_C++/socket.htm)
    - Hint: Will need laptops for this…
- Tuesday / Thursday Labs
  - Sockets on the embedded platform
- Wednesday
  - RTOS Scheduling (Continued)

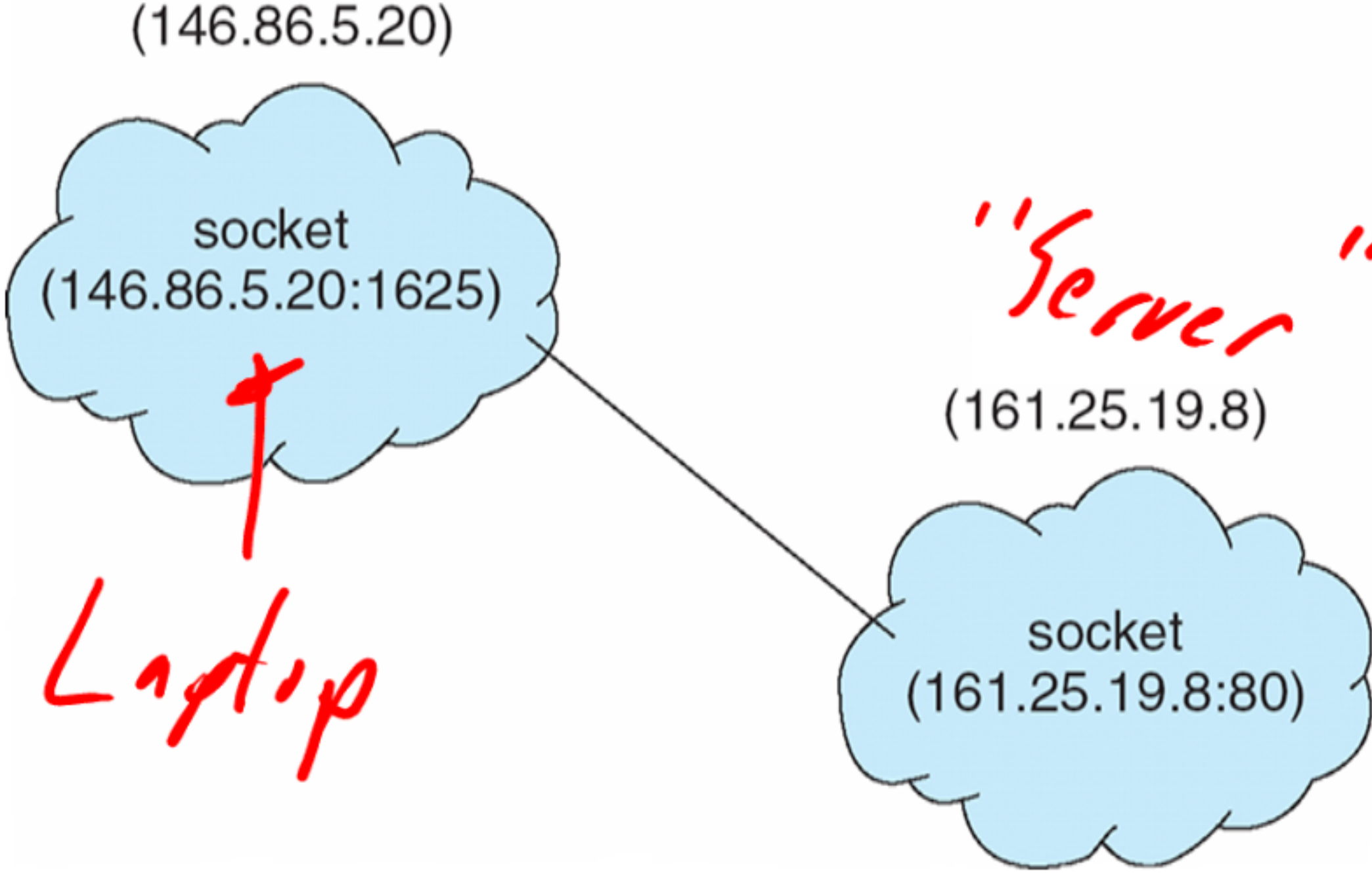
OBJECTIVES

- Understand the usage of sockets in a POSIX environment
- Construct a basic application using POSIX sockets

SOCKETS

- A socket is defined as an *endpoint for communication*
- Concatenation of IP address and port
  - Socket **161.25.19.8:1625**
  - Port **1625** ← 16 bits
  - Host **161.25.19.8** ← 32 bits
- Communication consists between a pair of sockets

SOCKET COMMUNICATION



PORT IDENTIFICATION

- 0 – 1023 Well known ports
  - 7 Echo
  - 20 ftp
  - 22 ssh
  - 25 smtp
  - 37 time
  - 70 gopher
  - 79 finger
  - 80 http
  - 666 doom
  - 992 telnet
- > 1024 Registered Ports
  - 1234 Mercurial / git
  - 1309 Altera Quartus
  - 1417 – 1420 Timbuktu Service
  - 1500 IBM Tivoli
  - 1534 Eclipse Agent Discovery
  - Etc

SOCKADDR_IN STRUCTURE

- **in_port_t**
  - An unsigned integral type of exactly 16 bits.
- **in_addr_t**
  - An unsigned integral type of exactly 32 bits.

port to connect to.

IP address

```
struct in_addr
{
        n_addr_t    s_addr;
}
```

```
struct sockaddr_in
{
short sin_family; /* must be AF_INET */
u_short sin_port;
struct in_addr sin_addr;
char sin_zero[8]; /* Not used, must be zero */
};
```

CREATING A NEW SOCKET

- The socket() system call creates a new socket.
  - 3 arguments
    - Address domain of the socket.
      - AF_UNIX or AF_INET
    - Type of socket.
      - SOCK_STREAM or SOCK_DGRAM.
    - Protocol.
      - Usually 0

BINDING TO A SOCKET

- bind() system call binds a socket to an address
- Takes three arguments
  - socket file descriptor
  - address to which is bound
  - size of the address to which it is bound.

- Listen()
  - Allows the process to listen for connections to a given socket
  - Two arguments
    - Socket file descriptor
    - Number of queues that can be waiting
- Accept()
  - Blocks until a client connects to a server

LISTENING FOR DATA

Server

Waiting for a connection

READING AND WRITING DATA

- read() attempts to read nbyte bytes of data from the object referenced by the descriptor fd into the buffer pointed to by buf.

- Write()
  - Writes data to the buffer

Same as File I/O in C.

LETS LOOK AT AN EXAMPLE

- We'll look at a program which has two pieces
  - Server
    - Receives ASCII text message
    - Prints it out to the console
    - Converts it to upper case
    - Sends it back to the client
  - Client
    - Prompts the user to enter a message
    - Sends it to the server
    - Prints out the response messages