Computer Networks

Adrian Sergiu DARABANT

Lecture 1

Introduction - Administrative

- Weekly lectures + lab
- Final grade:
  - Final written examination
  - Labs
  - Practical exam
    - Don’t know yet – it depends on your lab activity ☺
- Prerequisites
  - C/C++ system programming (Unix and Windows)
  - Operating systems

Bibliography

1. A.S. Tanenbaum – *Computer Networks 4th ed.*, Prentice Hall, 2003
2. J. Kurose, K. Ross, *Computer Networking: A Top Down Approach*, Addison-Wesley, rev2,3,4 2002-2007.
3. Douglas E. Comer, Internetworking with TCP/IP
    1. Vol 1- Principles, Protocols, and Architecture
    2. Vol 3- Client-Server Programming and Applications
4. G.R.Wright, R. Stevens, TCP/IP Illustrated – vol 1,2, Addison Wesley.
5. Matt Naugle, Illustrated TCP/IP – A Graphic Guide to protocol suite, John Willey & Sons, 1999.
6. W. Richard Stevens, Bill Fenner, Andrew M. Rudoff, *UNIX® Network Programming Volume 1, Third Edition: The Sockets Networking API*

Course Information

http://www.cs.ubbcluj.ro/~dadi/compnet

Required (?!) Tools/Materials

- Windows 32/64 – Development Env !
- VMware Player/ Virtual Box, etc
  - Install Linux / Windows !
  - Integration Tools
  - **Development Environment** Or **Vi** ?!?
- Set networking as *bridged* on VM!

You will thank me later :P

Syllabus

- **Communication basics**
  - Media and signals
  - Asynchronous and synchronous communication
  - Relationship among bandwidth, throughput, and noise
  - Frequency-division and time-division multiplexing

Sylabus-2

- **Networking and network technologies**
  - Packing switching
  - Framing, parity, and error detection
  - Local and wide area technologies
  - Network addressing
  - Connection, wiring and extension (repeaters, bridges, hubs, switches)
  - Forwarding and measuring of delay and throughput
  - Protocol layers

Syllabus-3

- **Internets and Internetworking**
  - Motivation and concept
  - Internet Protocol (IP) datagram format and addressing
  - Internet routers and routing
  - Address binding (ARP)
  - Internet Control Message Protocol (ICMP)
  - User Datagram Protocol (UDP)
  - Transmission Control Protocol (TCP)
  - Network Security

Syllabus-4

- **Network Applications**
  - Domain Name System (DNS)
  - File Transfer Protocol (FTP)
  - Remote Login Protocol (TELNET)
  - Email Transfer (SMTP)
  - Web technologies and protocol (HTTP)

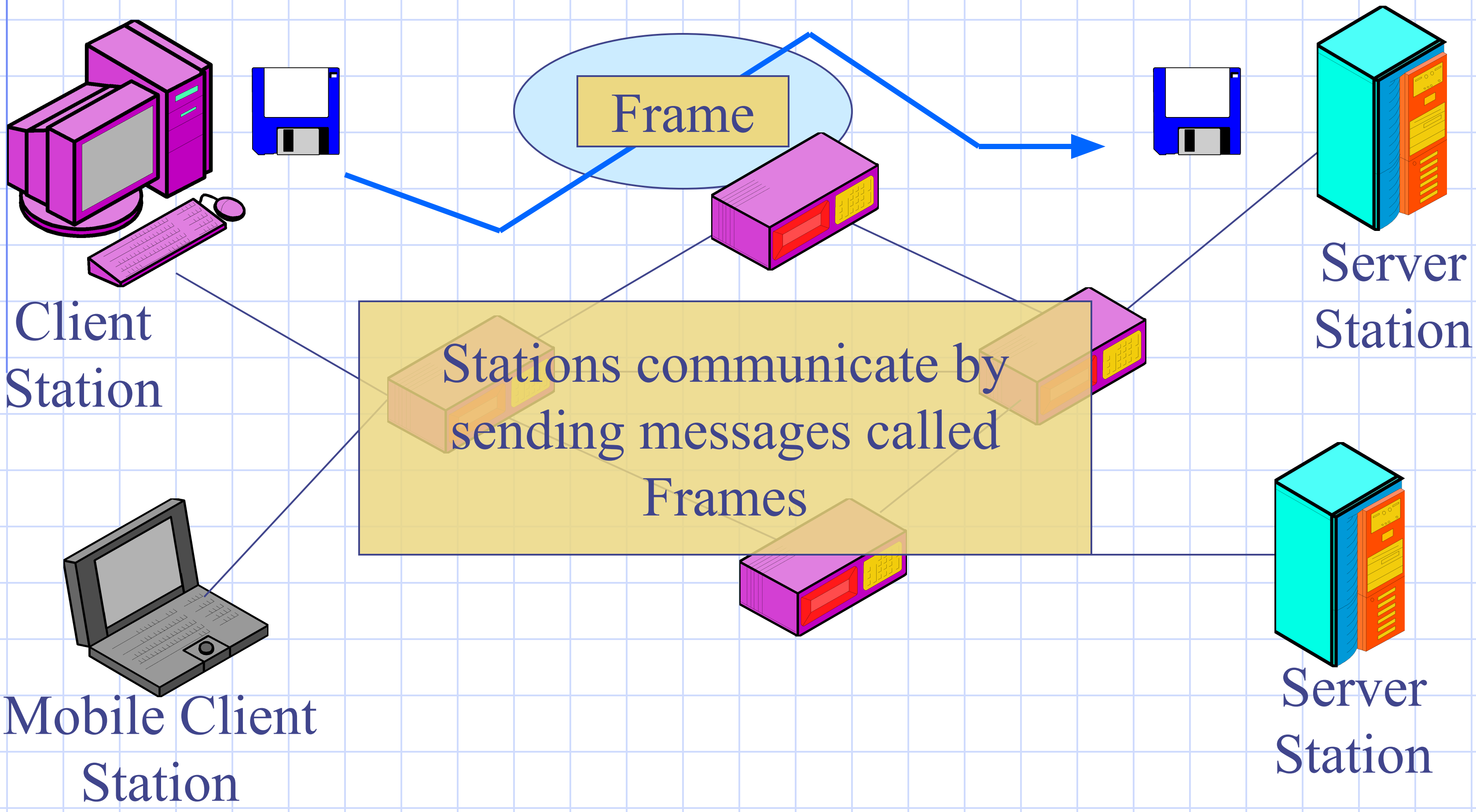
What is a Computer Network ?

- A collection of computers (PCs, Workstations) and other devices interconnected.
- **Components**:
  - Hosts (computers)
  - Links (coaxial cable, twisted pair, optical fiber, radio, satellite)
  - Switches/routers (intermediate systems)

Major Network Categories

- The global Internet
- Internal corporate networks
- The worldwide telephone system

What is a Computer Network?



Frame

Client Station

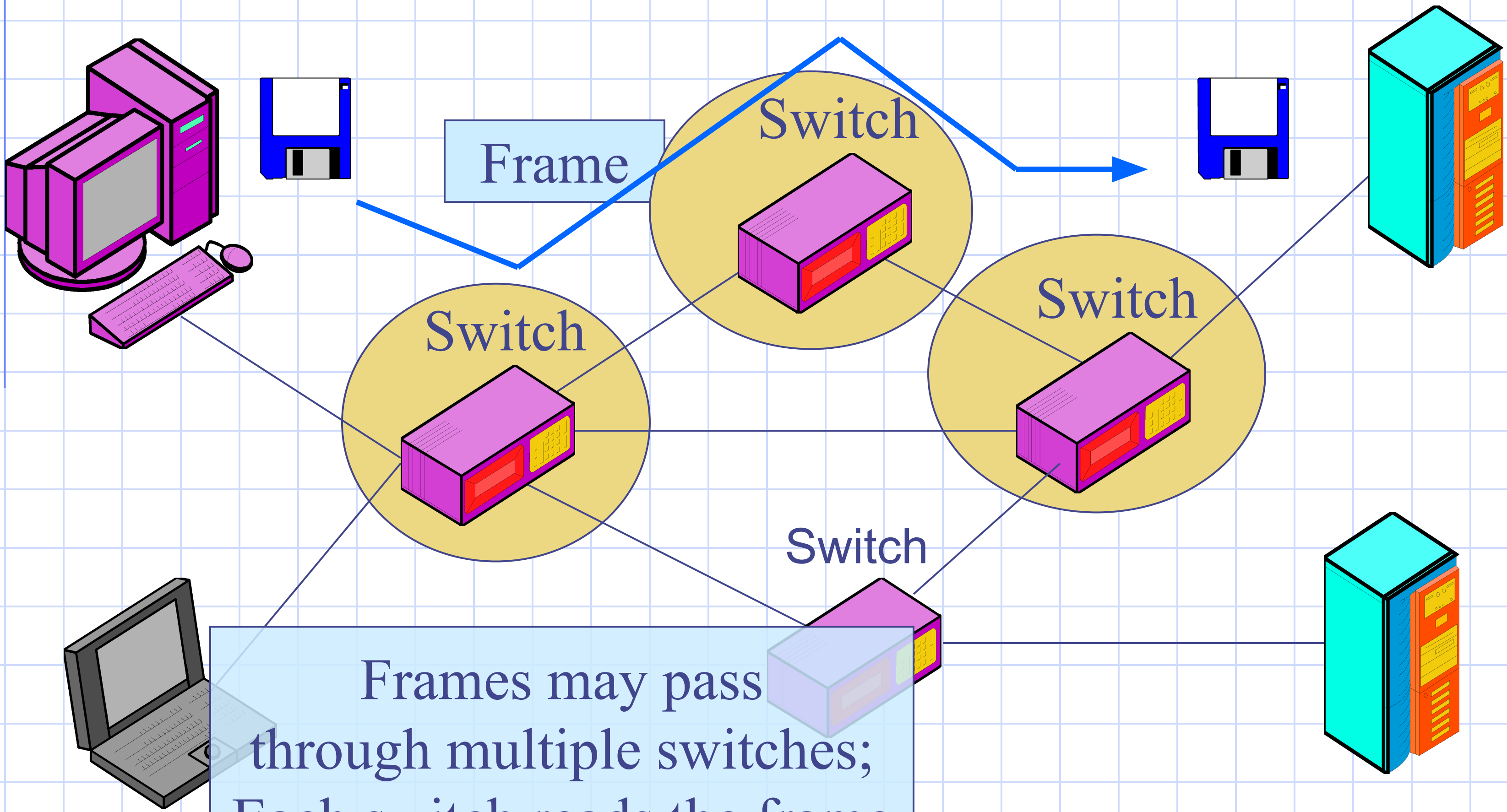
Server Station

Mobile Client Station

Server Station

Stations communicate by sending messages called Frames

What is a Computer Network?

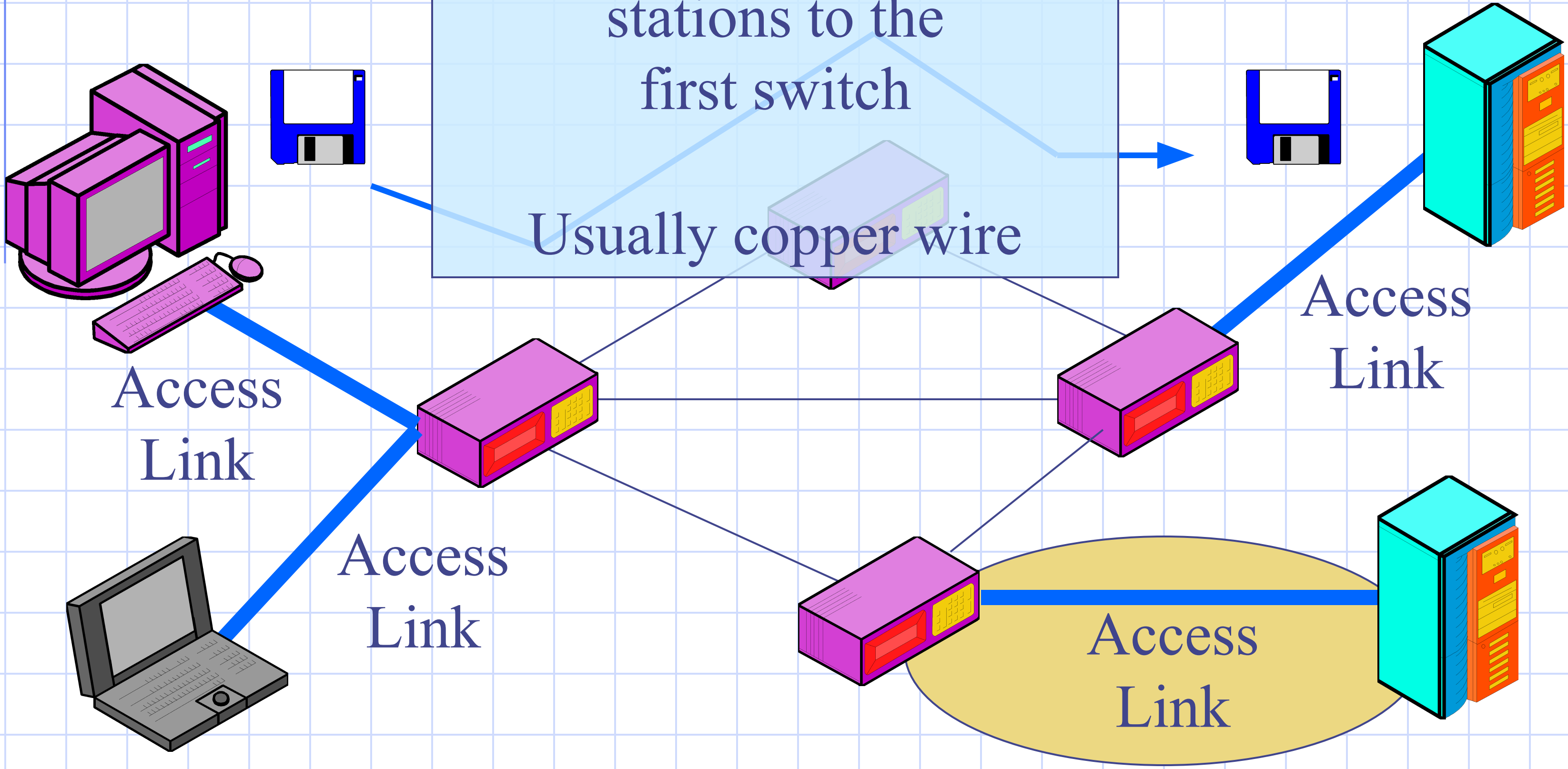


Frames may pass through multiple switches; Each switch reads the frame And passes it on

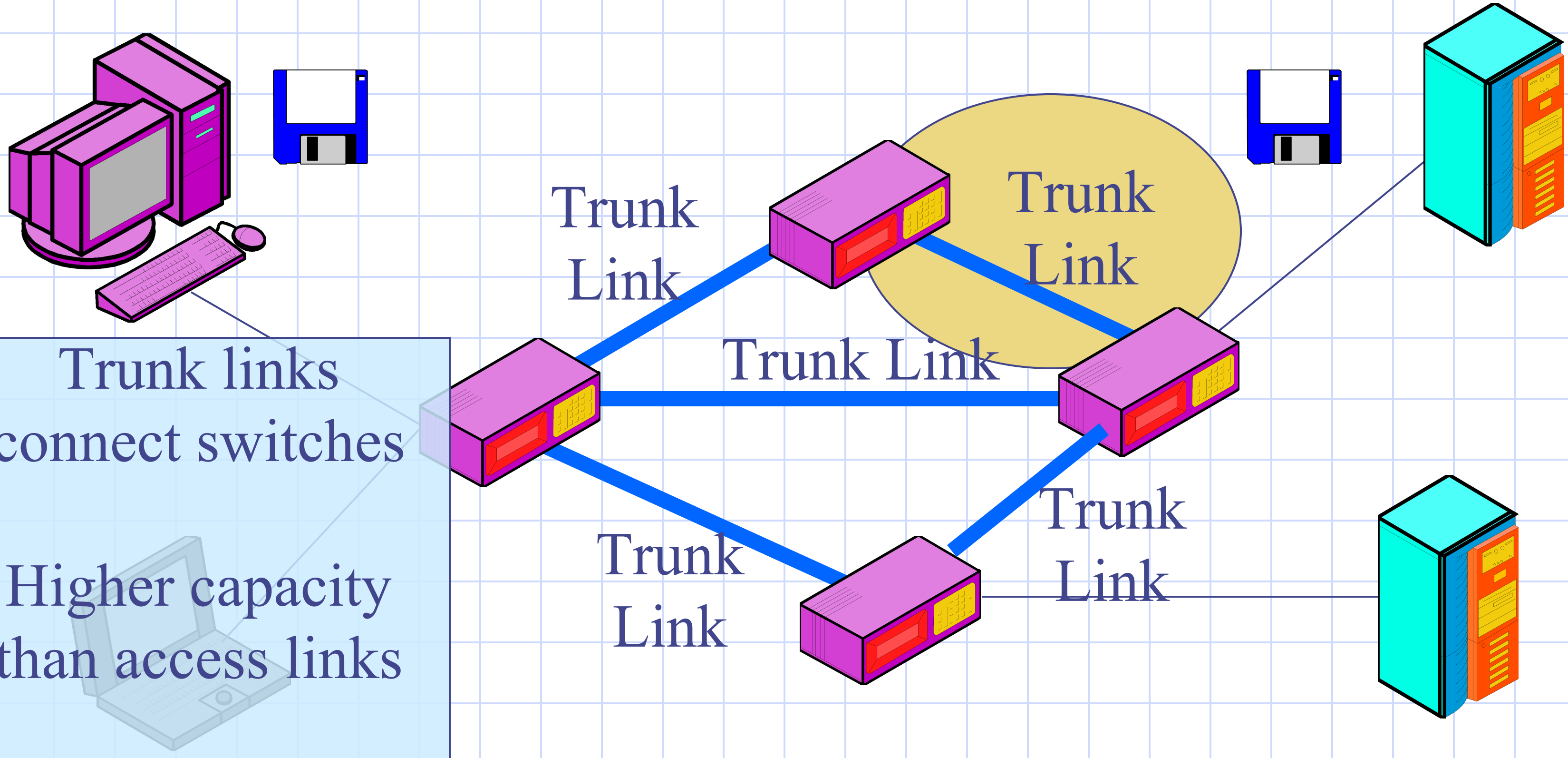
What is a Computer Network?

Access links connect stations to the first switch

Usually copper wire



What is a Computer Network?



Trunk links connect switches

Higher capacity than access links

Often optical fiber

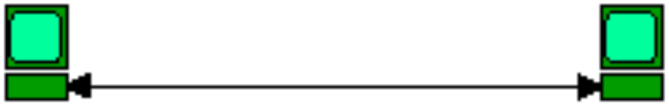
Classifications

1. Types of links
    - Direct links
    - Bus type links

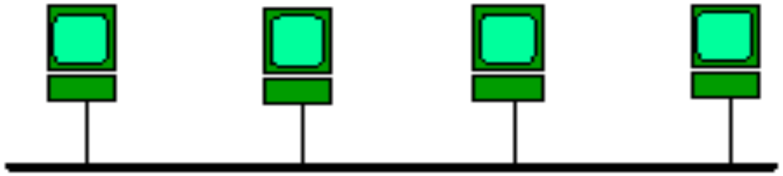
- Type of transmission
    - Circuit switched networks
    - Packet switched networks
    - Frame Relay
    - Asynchronous Transfer Mode (ATM)

Types of communication

1. **Types of links (connectivity)**



Direct -Point-to-point communication

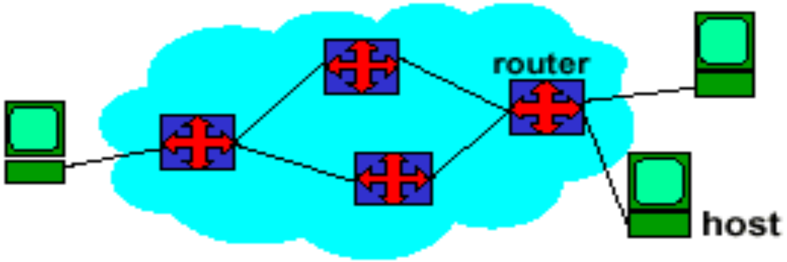


Direct - BUS Type / Multiple-access

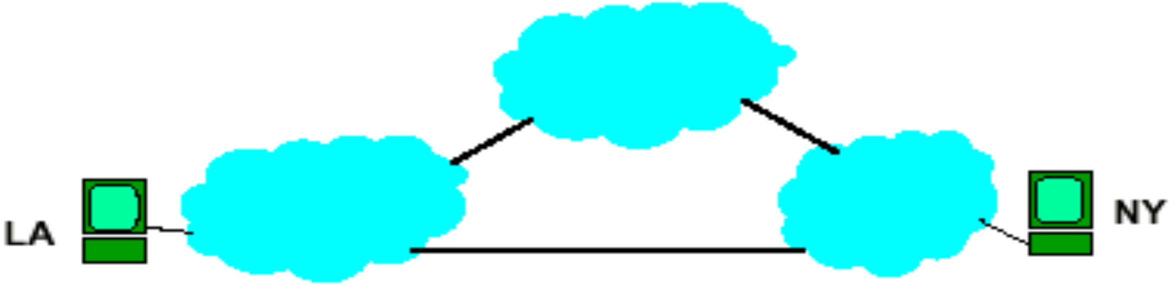
Types of Communication

2. Switched Networks

- Circuit - switched network: public telephone network

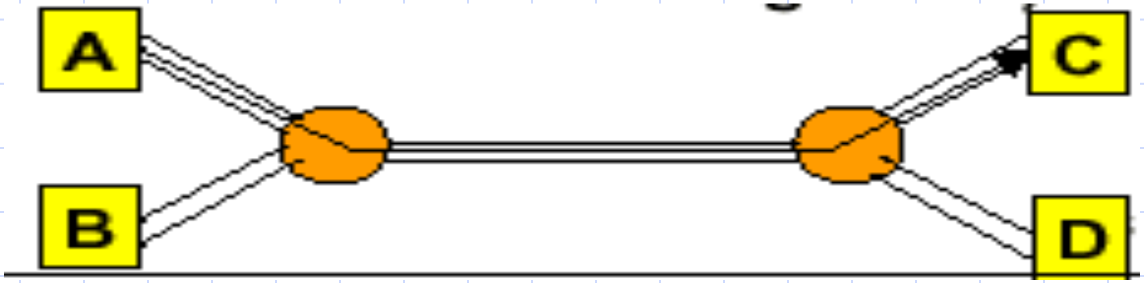


- Packet switched network: Internet (collection of networks)



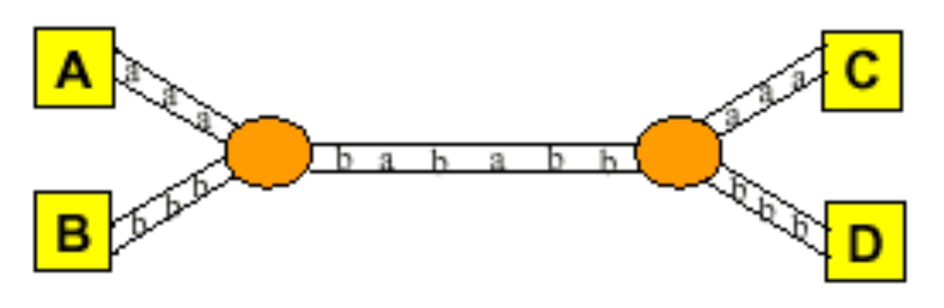
Circuit-Switching

- Set up a connection path (circuit) between the source and the destination (permanent for the lifetime of the connection)
- All bytes follow the same dedicated path
- Used in telephony
- Advantages: dedicated resources
- Disadvantages: not very efficient (lower utilization, e.g., a person talks < 35% of the time during a call)
- While A talks to C, B cannot talk to D on the same line.



Packet-Switching

- Packets from different sources are interleaved



- Efficient use of resources (since they are used on a demand): statistical multiplexing. Nobody reserves a lane on a freeway.
- Can accommodate bursty traffic (as opposed to circuit-switching where transmission is at constant rate).

Types of Communication

- Frame Relay
  - Alternative for Packet switching systems
  - Packet switching have large overheads to compensate for errors.
- ATM
  - Asynchronous Transfer Mode
  - Evolution of Frame Relay
  - Little overhead for error control
  - Fixed packet length

Communication infrastructure - Goals

- Reliable data delivery
- Error free data transmission
- Messages delivered in the same order they where sent
- Minimum guaranteed throughput
- Limited maximum delay
- Confidentiality
- Authentification

Network programming

- Programmer does not need to understand the hardware part of network technologies.
- Network facilities accessed through an *Application Program Interface* - API
- Communication
  - Connection oriented
  - Datagram Oriented

Connection oriented-API

- The BSD socket library
  - Socket
  - Bind
  - Listen, Accept
  - Connect
  - Read, Write, Recv, Send
  - Close, Shutdown
- Where do we get info on these ?
  - man, msdn

Socket Example

Server.c

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <stdio.h>
#include <unistd.h> /* close */

#define SERVER_PORT 1500
```

```
int main (int argc, char *argv[]) {
  int sd, newSd, cliLen;
  struct sockaddr_in cliAddr, servAddr;
  char line[MAX_MSG];
  int len;

  sd = socket(AF_INET, SOCK_STREAM, 0);
   if(sd<0) {
    perror("cannot open socket ");
    return ERROR;
  }

  /* bind server port */
  servAddr.sin_family = AF_INET;
  servAddr.sin_addr.s_addr = htonl(INADDR_ANY);
  servAddr.sin_port = htons(SERVER_PORT);
```

```
if (bind(sd, (struct sockaddr *)
     &servAddr, sizeof(servAddr))<0) {
   perror("cannot bind port ");
   return ERROR;
 }
 listen(sd,5);
 while(1) {
    printf("%s: waiting for data on port
     TCP %u\n",argv[0],SERVER_PORT);

   cliLen = sizeof(cliAddr);
   newSd = accept(sd, (struct sockaddr *)
     &cliAddr, &cliLen);
   if(newSd<0) {
     perror("cannot accept connection ");
     return ERROR;
    } // end if
```

```
  /* init line */
   memset(line,0,MAX_MSG);

   /* receive segments */
  if ( (len=read(newSd,line,MAX_MSG))> 0) {
    printf("%s: received from %s:TCP%d : %
    s\n", argv[0],
           inet_ntoa(cliAddr.sin_addr),
           ntohs(cliAddr.sin_port), line);


     write(newSd,line,len);
   } else
     printf("Error receiving data\n");
   close(newSd);
  } //end if
} //end while
```

CLIENT.C

```
include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <stdio.h>
#include <unistd.h> /* close */

#define SERVER_PORT 1500
#define MAX_MSG 100

int main (int argc, char *argv[]) {

  int sd, rc, i;
  struct sockaddr_in servAddr;
  struct hostent *h;
  char msg[300];
```

```
 if(argc < 3) {
  printf("usage: %s <server> <text>\n",argv[0]);
  exit(1);
}

h = gethostbyname(argv[1]);
if (h==NULL) {
  printf("%s: unknown host '%s'\n",argv[0],argv
    [1]);
  exit(1);
}

servAddr.sin_family = h->h_addrtype;
memcpy((char *) &servAddr.sin_addr.s_addr,
    h->h_addr_list[0], h->h_length);
servAddr.sin_port = htons(SERVER_PORT);
```

```
/* create socket */
  sd = socket(AF_INET, SOCK_STREAM, 0);
  if(sd<0) {
    perror("cannot open socket ");
    exit(1);
  }
  /* connect to server */
  rc = connect(sd, (struct sockaddr *) &servAddr, sizeof(servAddr));
  if(rc<0) {
    perror("cannot connect ");
    exit(1);
  }
  write(rc, argv[1],strlen(argv[1]+1) );
  read(rc, msg, 300);
  printf(”Received back: %s\n”, msg);
  close(rc);
  return 0;
}
```