Computer Networks. Unit 1: Introduction

Notes of the subject *Xarxes de Computadors, Facultat Informàtica de Barcelona, FIB*

Llorenç Cerdà-Alabern

February 13, 2019

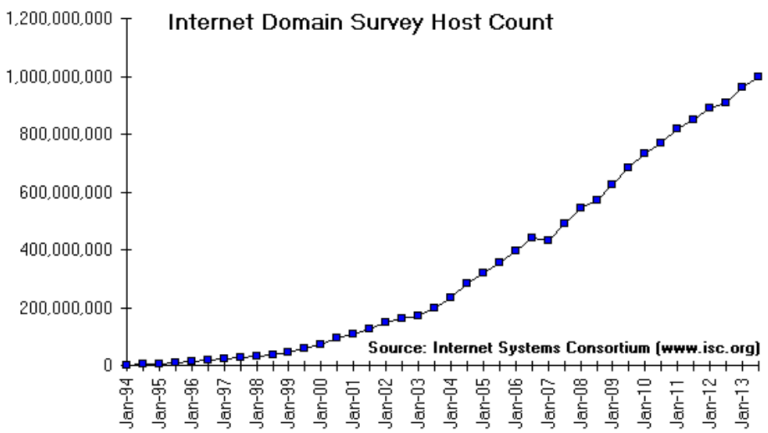
Contents

1 Unit 1: Introduction

1.1 What is a Computer Network?

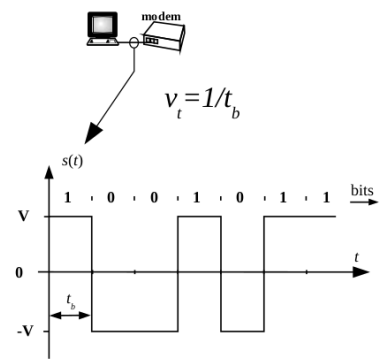
Brief history:

- **1830** Telegraph
- **1875** Alexander Graham Bell patent the telephone
- **1951** First commercial computer
- **1960** ARPANET. Public networks rediris geant
- **1972** First International and commercial Packet Switching Network, X.25
- **1990** The Internet is opened to the general public



1.2 Bits per second (bps)

- **line bitrate**



NRZ signal

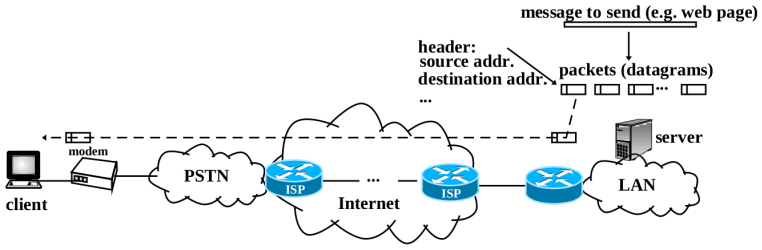
- **throughput** (velocidad efectiva)

$$v_{ef}[\text{bps}] = \frac{\text{number of information bits}}{\text{observation time}}$$

- Prefixes:
  - **k**, kilo: 10^3
  - **M**, Mega: 10^6
  - **G**, Giga: 10^9
  - **T**, Tera: 10^{12}
  - **P**, Peta: 10^{15}

1.3 Packet switching URL

- **Virtual Circuit**: Connection oriented, used in WANs, e.g. X.25, Frame Relay, ATM.
- **Datagram**: Connectionless, used in the Internet.



Datagram packet switching

PSTN: Public Switched Telephone network
WAN: Wide Area Network
LAN: Local Area Network
ATM: Asynchronous Transfer Mode

1.4 Standardization Bodies

1. *Int. Telecommunication Union*, **ITU**
   - WAN standards. URL
2. *Int. Organization for Standardization*, **ISO**
   - Industrial standards. URL.
3. *Institute of Electrical and Electronics Engineers*, **IEEE**
   - LAN standards. URL.
4. *European Telecommunications Standards Institute*, **ETSI**
   - Mobile phone standards (GSM). URL.

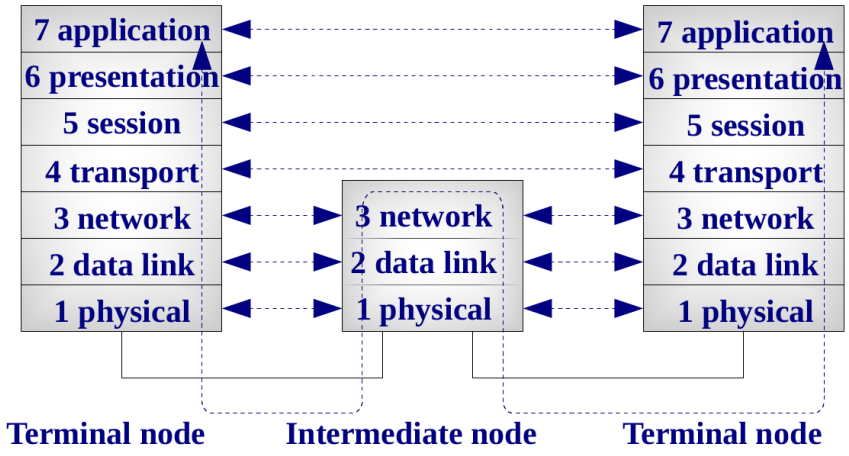
5. *Telecommunications Industry Association*, **TIA**
    - Cabling standards. URL.
6. *World Wide Web Consortium*, **W3C**. URL

Internet:

1. *Internet Engineering Task Force*, **IETF**. URL.
    - Request For Comments, **RFCs**. URL

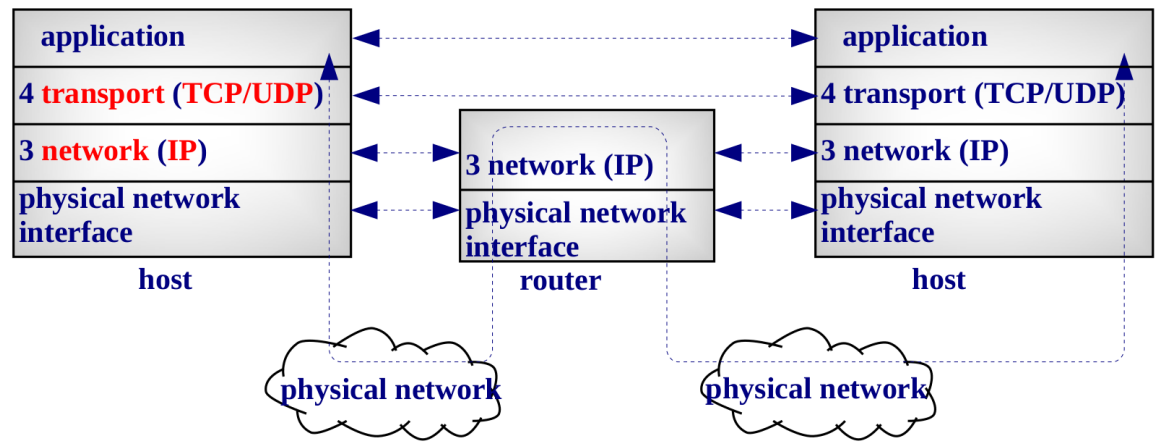
1.5 ISO OSI Reference Model URL

OSI: *Open Systems Interconnection*

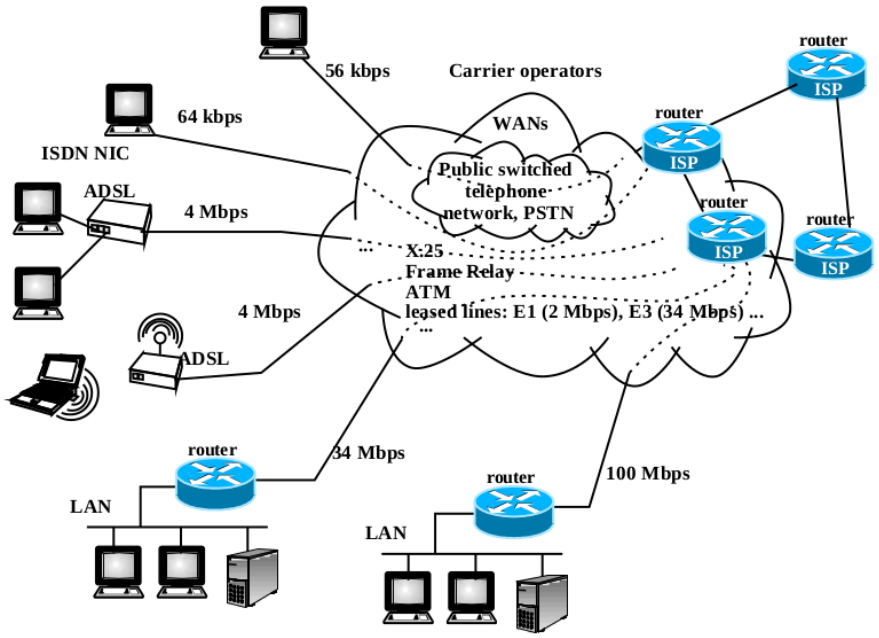


1.6 TCP/IP Architecture URL

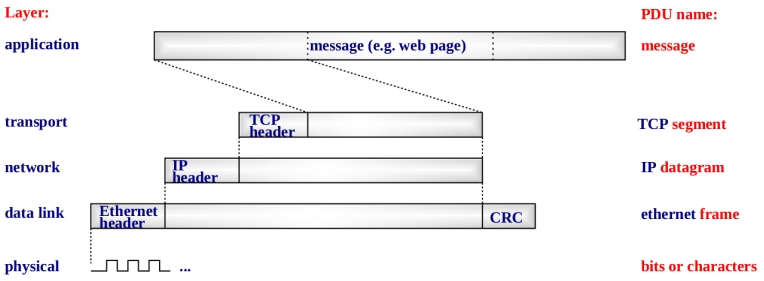
- No RFC specifies the TCP/IP model.



1.7 Internet Infrastucture URL



1.8 Encapsulation URL

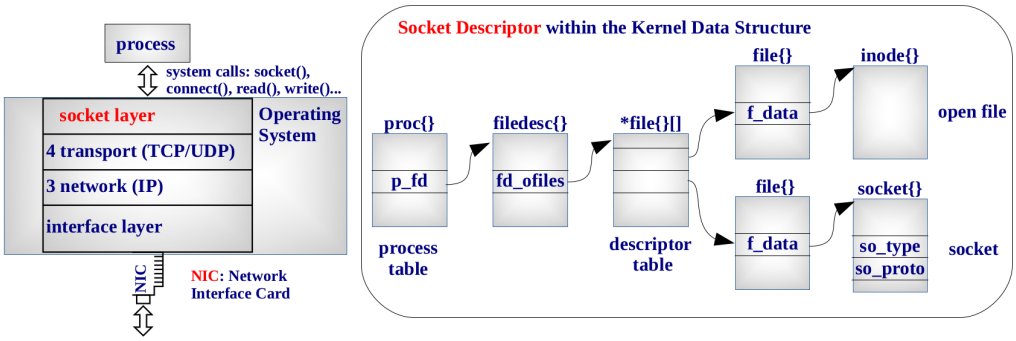


PDU: Protocol Data Unit
CRC: Cyclic Redundancy Check

Network sniffers (bash)

```
sudo tcpdump -ni wlan0 # command line sniffer
sudo wireshark         # graphical sniffer
```

1.9 TCP/IP Implementation URL



TCP and UDP sockets (bash)

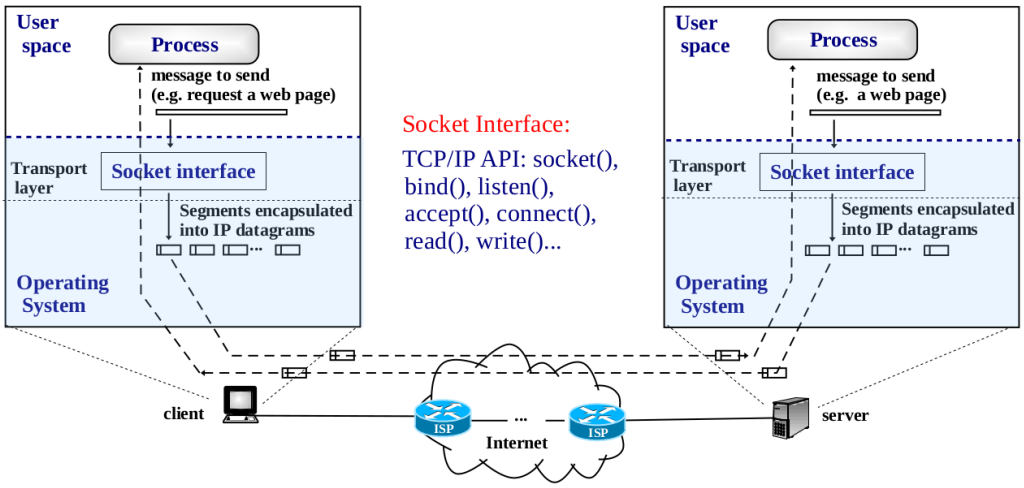
```
netstat -nt # list TCP sockets
netstat -nu # list UDP sockets
```

Sockets opened by a browser (bash)

```
netstat -nt
```

1.10 Client Server Paradigm URL

- The server "listens" a **well known port** (< 1024).
- The client connects with an **ephemeral port** (>=1024).



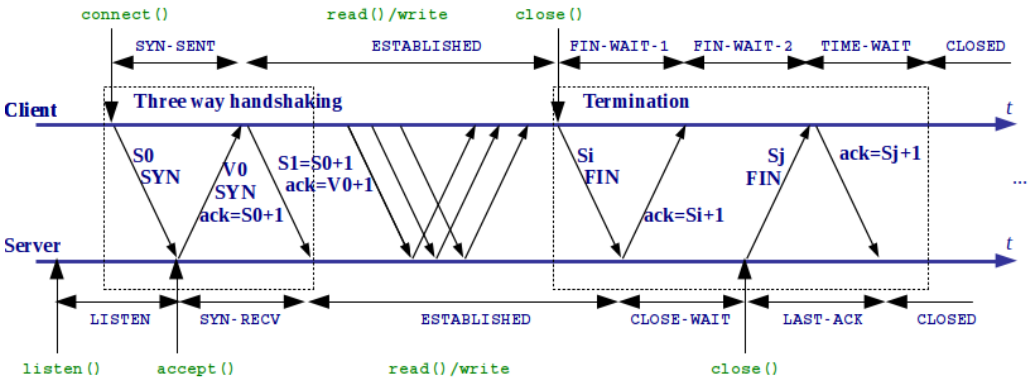
API: Application Programming Interface

TCP and UDP servers (bash)

```
netstat -nat
netstat -nau
file /etc/services
```

1.11 Transport layer: UDP/TCP

- UDP **User Datagram Protocol**: Connectionless, no reliable.
- TCP **Transmission Control Protocol**: Connection oriented, reliable.



1.12 Practical examples

tcpdump (bash)

```
tcpdump -ni lo
```

Minimal UDP server (perl)

```perl
#!/usr/bin/perl -w
use IO::Socket;
use strict;
use Data::Dumper;

my $sock = IO::Socket::INET->new(
    Proto     => 'udp',
    LocalPort => 5000 # server port
) or die "Could not create socket: $!\n";

my $MAXLEN = 1500 ;
my $newmsg ;
print "Awaiting UDP messages\n";
while ($sock->recv($newmsg, $MAXLEN)) {
    my($port, $ipaddr) = sockaddr_in($sock->peername);
    my $hishost = gethostbyaddr($ipaddr, AF_INET);
    print "Received from $hishost: $newmsg\n";
}
```

Minimal UDP client (perl)

```perl
#!/usr/bin/perl -w
use IO::Socket;
use strict;
use Data::Dumper;

my $sock = IO::Socket::INET->new(
    Proto    => 'udp',
    PeerPort => 5000, # server port
    PeerAddr => '127.0.0.1',
) or die "Could not create socket: $!\n";

(my $message = sprintf "%-50s", "1") =~ tr/ /1/;
print localtime() . ": sending " . substr($message, 0, 10) . " x " . length($message) . "\n" ;
$sock->send($message) or die "Send error: $!\n";
```

Minimal TCP server (perl)

```perl
#!/usr/bin/perl -w
use IO::Socket::INET; use Term::ANSIColor;

print "Sart TCP server.\n" ;
my $s_sock = IO::Socket::INET->new(
    LocalPort => 5000,
    Proto     => 'tcp',
    Listen    => 5
) or die "Could not create socket!\n";

while(1) {
  my $c_sock = $s_sock->accept() ;
  printf colored("Accepted: ", 'green')."%s, %s\n",
    $c_sock->peerhost(), $c_sock->peerport() ;
  while(<$c_sock>) {
    print "Received from Client: $_";
  }
  print colored("Connection closed", 'red')."\n" ;
}
```

Minimal TCP client (perl)

```perl
#!/usr/bin/perl -w
use IO::Socket::INET;

print "Sart TCP client.\n" ;
my $socket = IO::Socket::INET->new(
    PeerHost => '127.0.0.1',
    PeerPort  => 5000,
    Proto     => 'tcp'
) or die "Could not create socket: $!\n";

print "TCP Connected.\n" ;
while (<>) {
  print "sending $_" ;
  $socket->send($_);
}
```

1.13 List of Acronyms

API: Application Programming Interface
ATM: Asynchronous Transfer Mode
CRC: Cyclic Redundancy Check
IP: Internet Protocol
LAN: Local Area Network
NIC: Network Interface Card
PDU: Protocol Data Unit
PSTN: Public Switched Telephone network
TCP: Transmission Control Protocol
UDP: User Datagram Protocol
URL: Uniform Resource Locator
WAN: Wide Area Network