DTC561 - Massimo Di Pierro

Instructor: Massimo Di Pierro

Class mailing list:
tdc561_901@mailman.depaul.edu

To enroll in mailing list:
http://mailman.depaul.edu/mailman/listinfo/tdc561_901

Books

textbook (official):

UNIX Network Programming by W. R. Stevens (Prentice Hall)

A good reference book, hard to read

optional: Linux Socket Programming by W. W. Gay (QUE)

Simple code examples and good explanations

optional: Using TCP/IP by R. Shanmugam et al. (QUE)

No code but good explanation of protocols

optional: Internetworking with TCP/IP by D.E.Comer et al. (Prentice Hall)

optional: Beej's Guide to Network Programming

http://www.ecst.scuchico.edu/~beej/guide/net/

class roadmap / syllabus

Week 1: OSI, IP, TCP, UDP, arp, netstat, tcpdump, client, server

Week 2: TCP (no threads), socket, bind, listen, connect, accept

Week 3: Overview of protocols http, ftp, pop, imap, telnet, dns

Week 4: TCP servers with threads, and applications (http, ftp, etc.)

Week 5: Concurrent Servers: blocking, non blocking, signal driven, asynchronous, multiplexing

Week 6: Datagrams and applications

Week 7: Multiprotocol servers, multiservice servers, inetd

Week 8: Multicasting (mbone)

Week 9: XML-RPC, SOAP and web services

Week 10: (projects)

Low level Posix API are in C. Our focus, OOP and security.

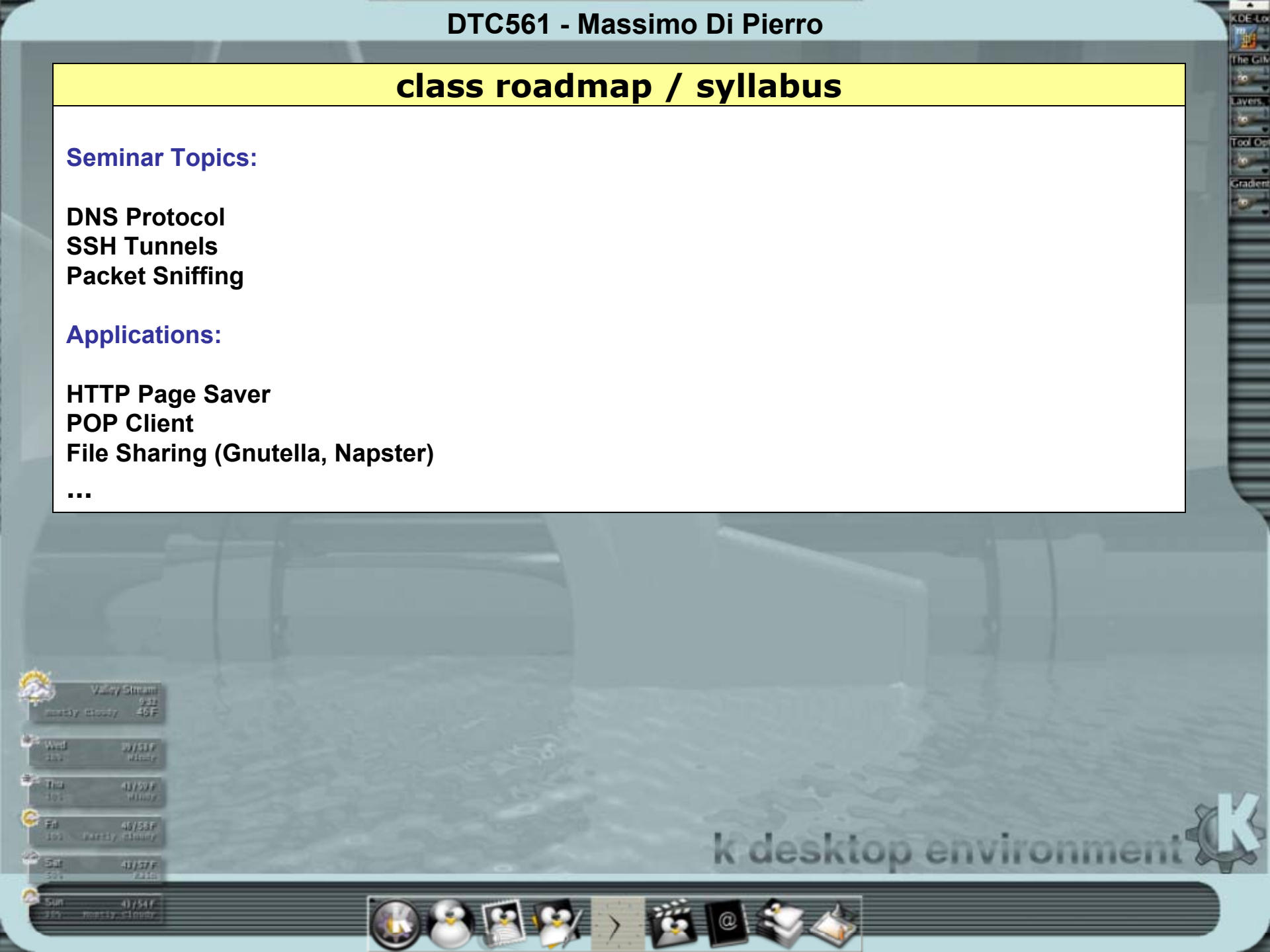
class roadmap / syllabus

Seminar Topics:

DNS Protocol
SSH Tunnels
Packet Sniffing

Applications:

HTTP Page Saver
POP Client
File Sharing (Gnutella, Napster)
...



class roadmap / unix, posix, protocols

UNIX COMMANDS:
bash
ls,cd,rm,mv,mkdir
rmdir,echo,>,<,&
ps, top, kill

UNIX NET COMMANDS:
ifup [eth0]
ifconfig
netstat
tdcdump -n -q
arp
ping [ip]
traceroute [ip]
curl (~)

POSIX STRUCTURES:
sockaddr
sockaddr_in
in_addr
hostent
timeval

POSIX API:
accept
close
connect
exec
fcntl
fork
freeaddressinfo
gethostbyaddr
gethostbyname
gethostname
Getpeername
getsockopt
getpid
htonl
htons
inet_addr
inet_aton
inet_ntoa
inet_ntop
inet_pton
ioctl
listen
memcmp
Memcpy
memset
ntohl
ntohs
open
poll
select
pthread_create
pthread_detach
pthread_exit
pthread_join
pthread_mutex_lock
pthread_mutex_unlock
read
recv
recvfrom
send
sendto
select
setsockopt
setpid
signal
socket
socketpair
system
wait
waitpid
writeg

NETWORK TYPES:
Ethernet (~)
Token Ring (~)

DATA LINK:
ARP / RARP

INTERNET:
IP

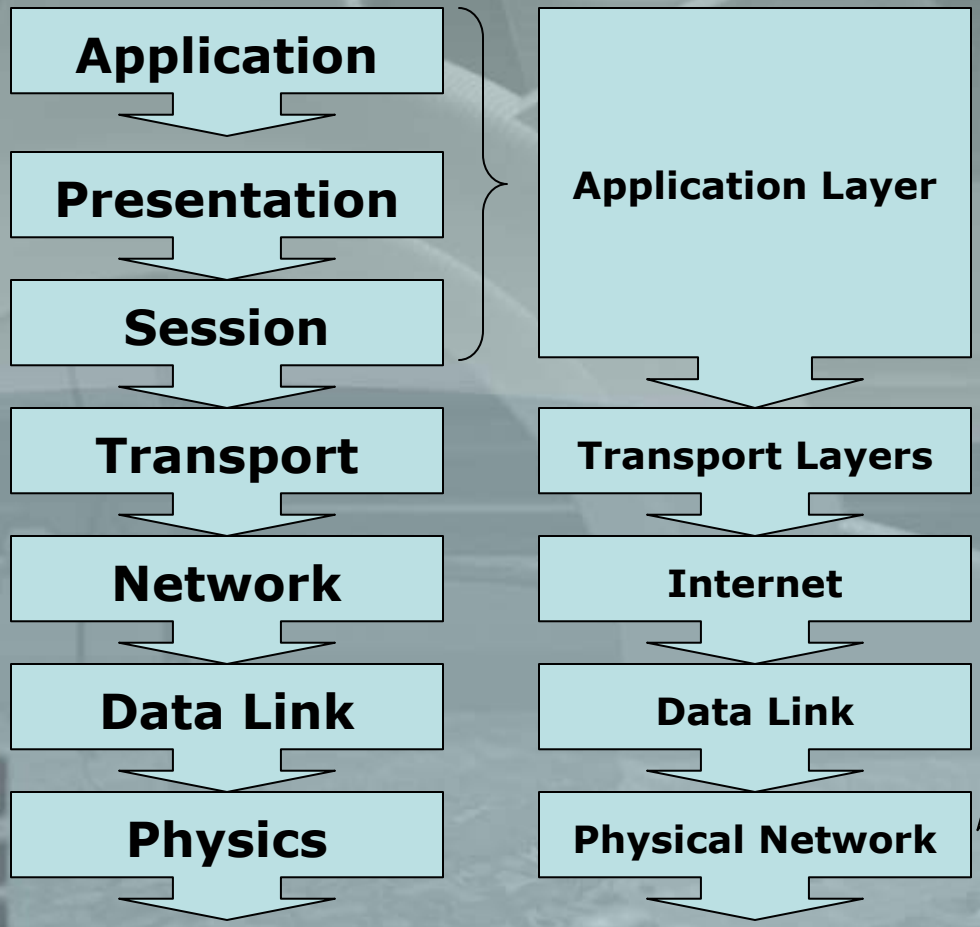
TRANSPORT:
TCP / UDP

APPLICATIONS:
http
ftp
telnet
pop / imap
dns
gnutella/napster
(http://rcf.net)

RPC:
xml-rpc
SOAP (gSOAP)
CORBA (~)

Referece

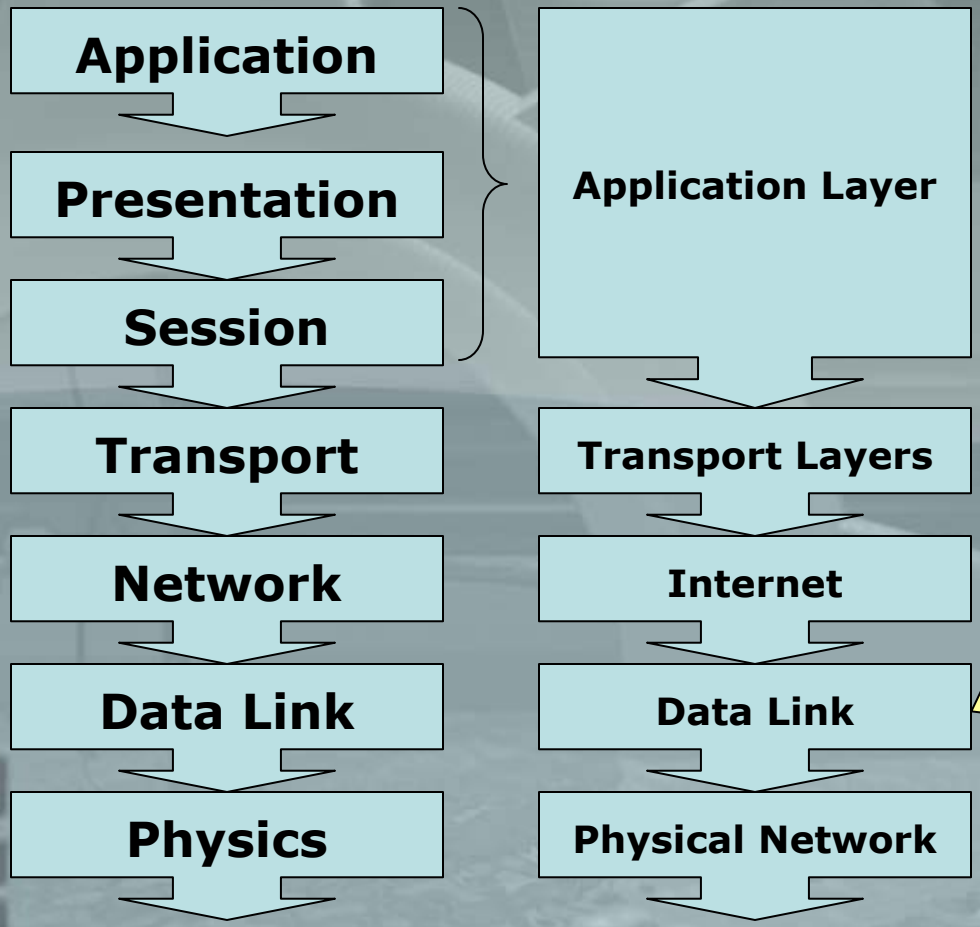
International Standard Organization's Open System Interconnection model



Ethernet:
(by Xerox Corp.)
A node send signal when network is free. Requires conflict resolution.

Token Ring:
A node get token, send signal, then release token.

International Standard Organization's Open System Interconnection model

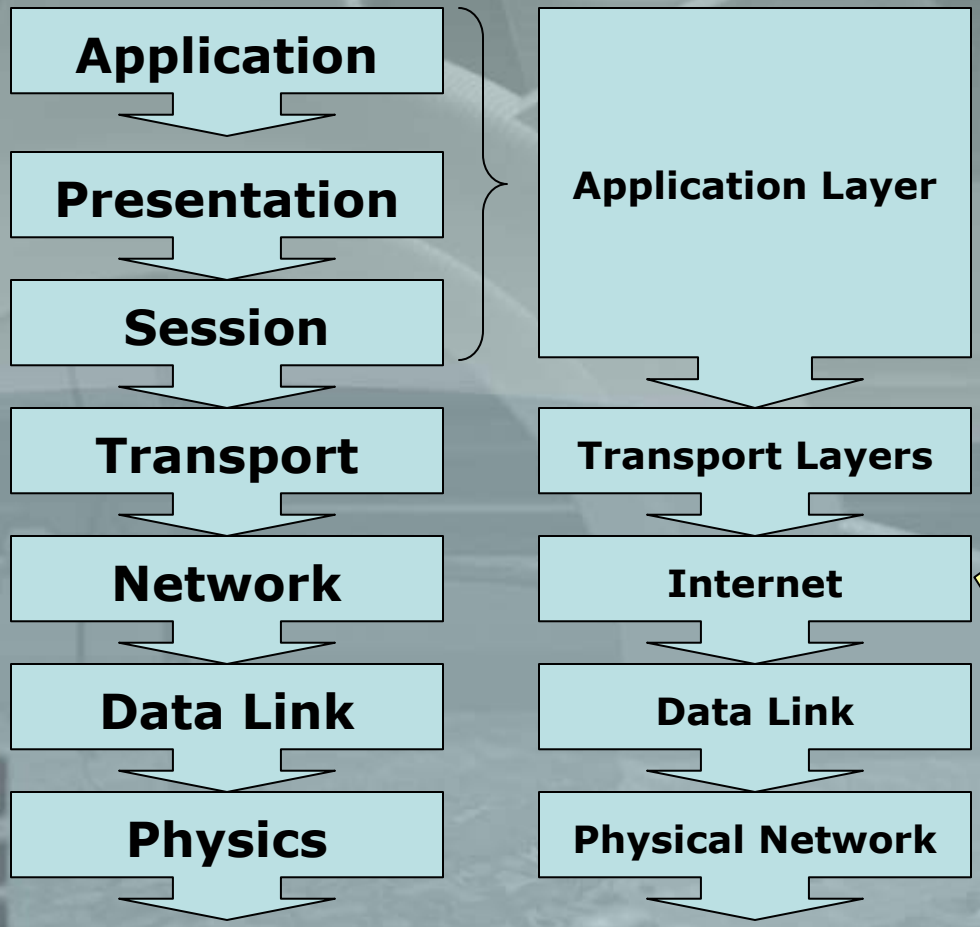


ARP:
(Address Resolution Protocol)

Converts IP address to MAC address

RARP:
(Reverse ARP)

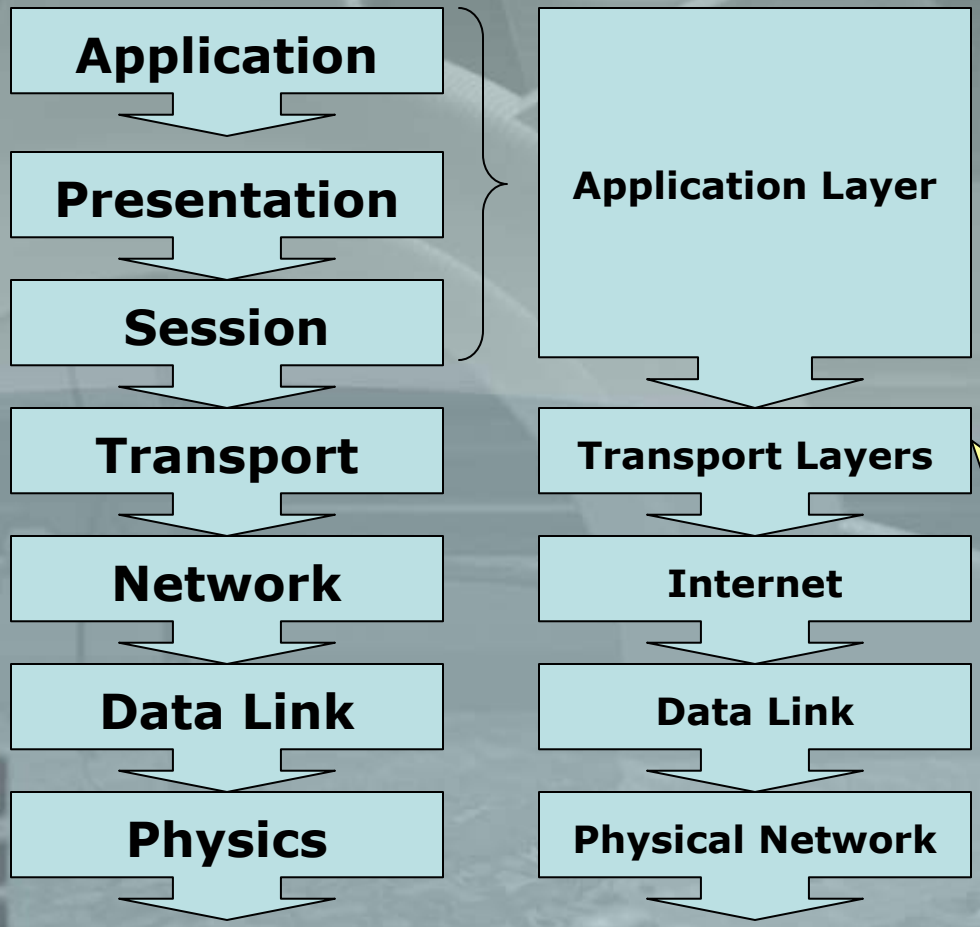
International Standard Organization's Open System Interconnection model



IP:
(Internet Protocol)
xxx.xxx.xxx.xxx

ICMP:
(Internet Control Message Protocol)

International Standard Organization's Open System Interconnection model



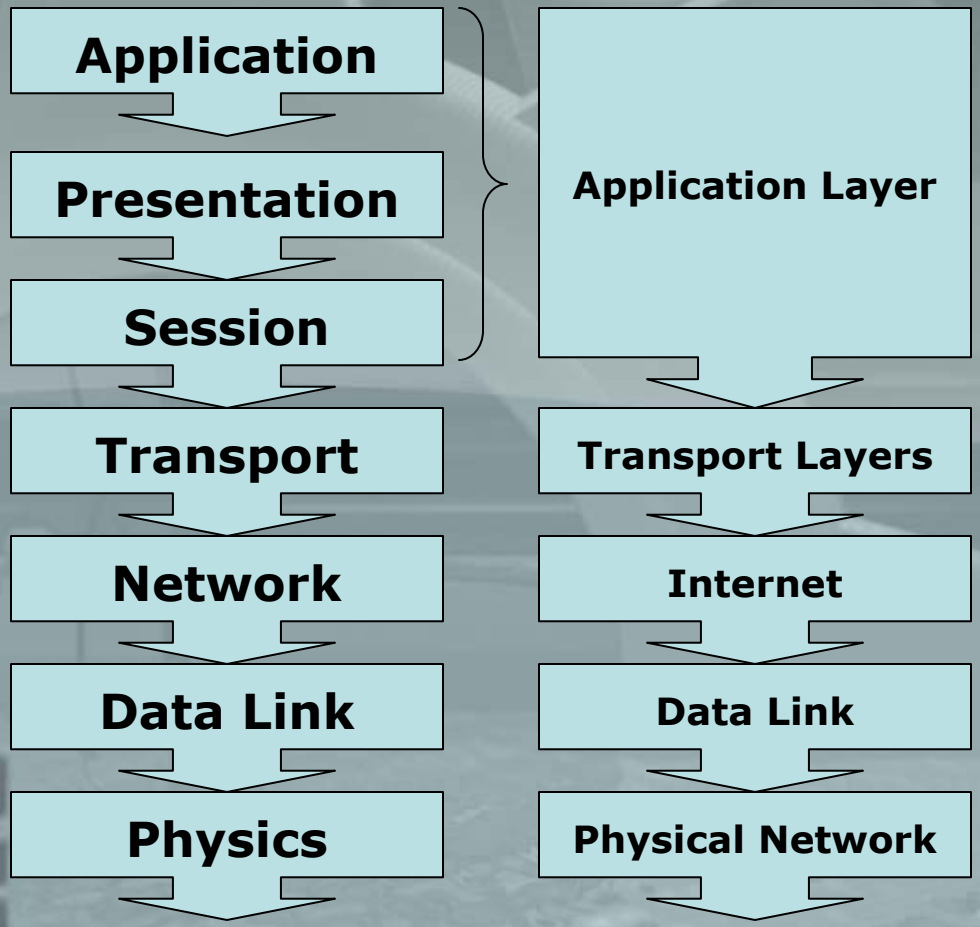
TCP:

A send message to B and B acknowledges receiving message. If not action is taken.

UDP:

A send message to B but no acknowledge. Messages may be lost.

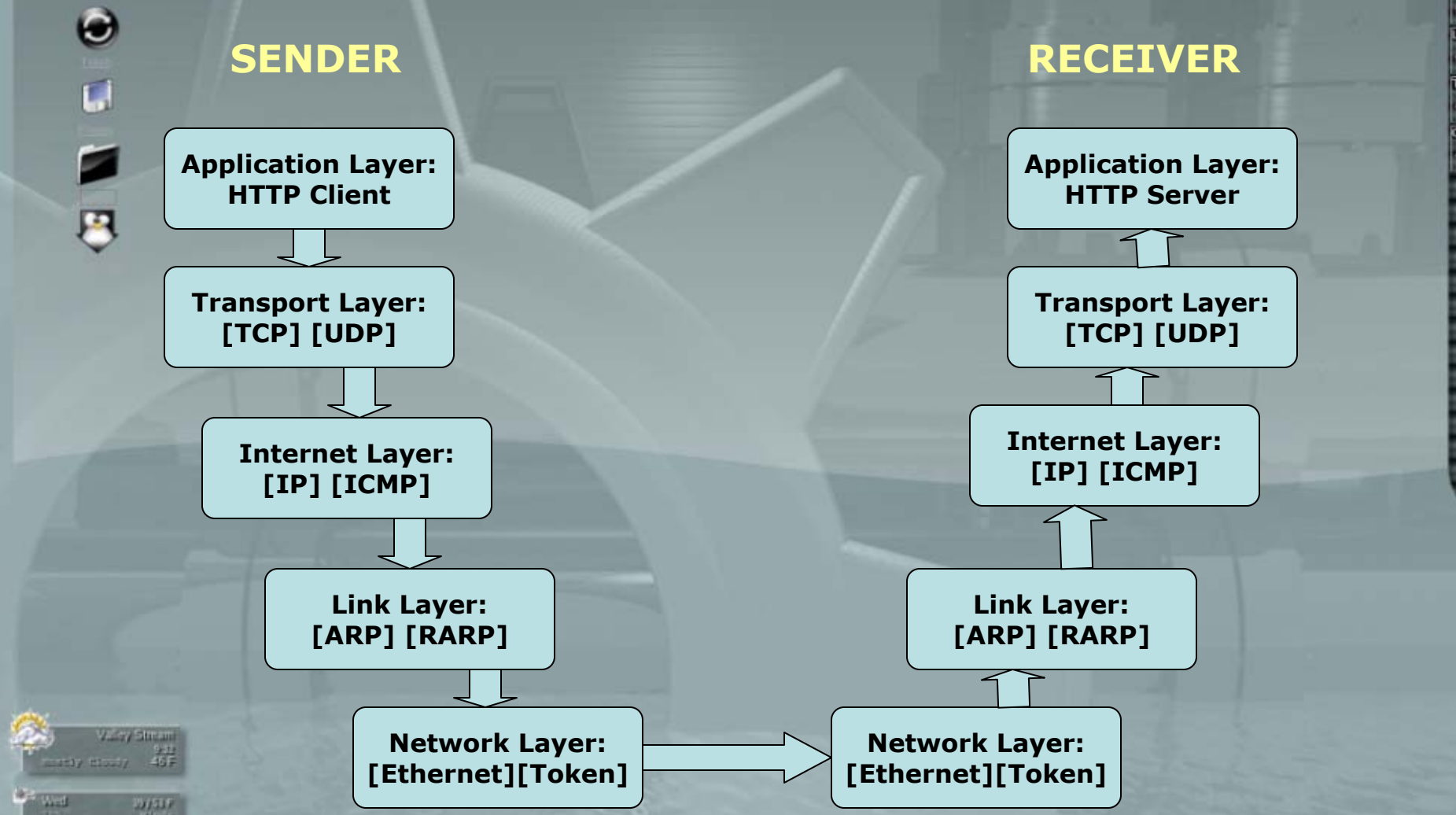
International Standard Organization's Open System Interconnection model



Examples:

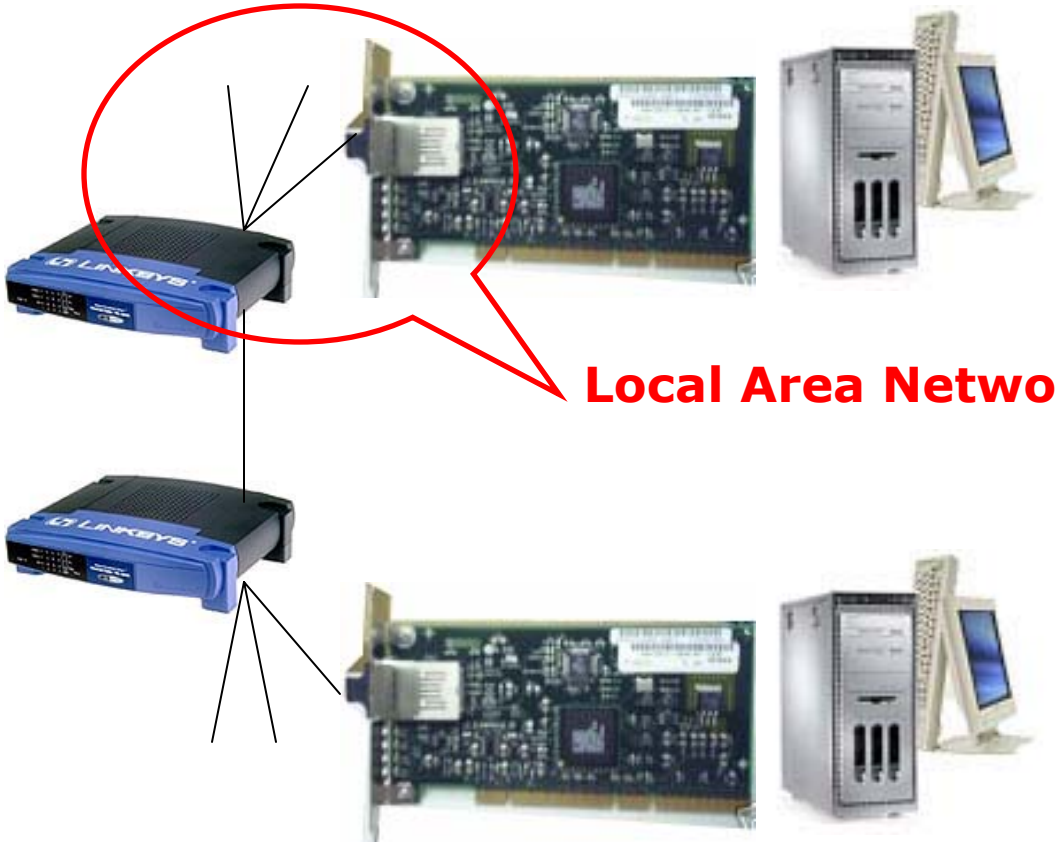
1) HTTP
2) Telnet
3) ssh
4) FTP
5) POP
6) SMTP
7) DNS
8) Napster
9) Gnutella
10) voiceIP
...

TDC561 - Massimo Di Pierro



- **Application Layer: HTTP, Telnet, etc.**
- **Transport Layer: [TCP] [UDP]**
- **Internet Layer: [IP] [ICMP]**
- **Link Layer: [ARP] [RARP]**
- **Network Layer: [Ethernet][Token]**

LAN / WAN



Application Layer:
HTTP, Telnet, etc.

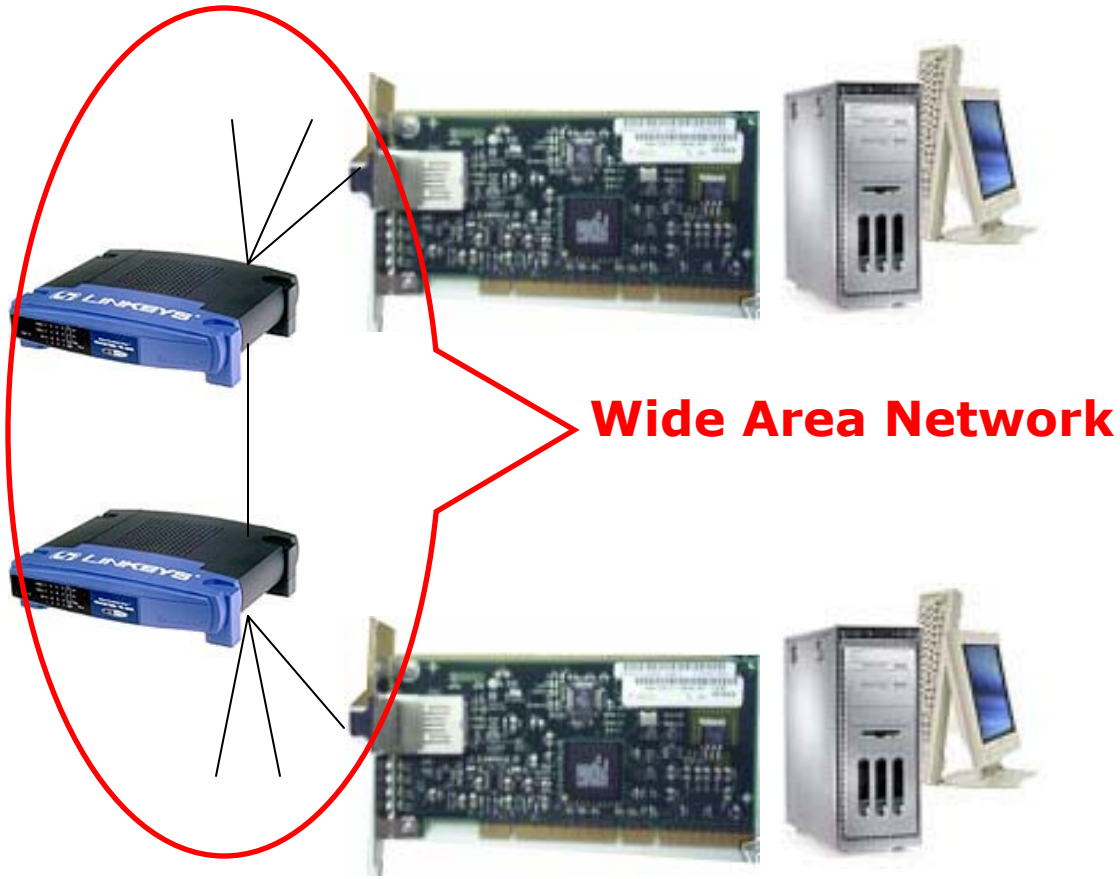
Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

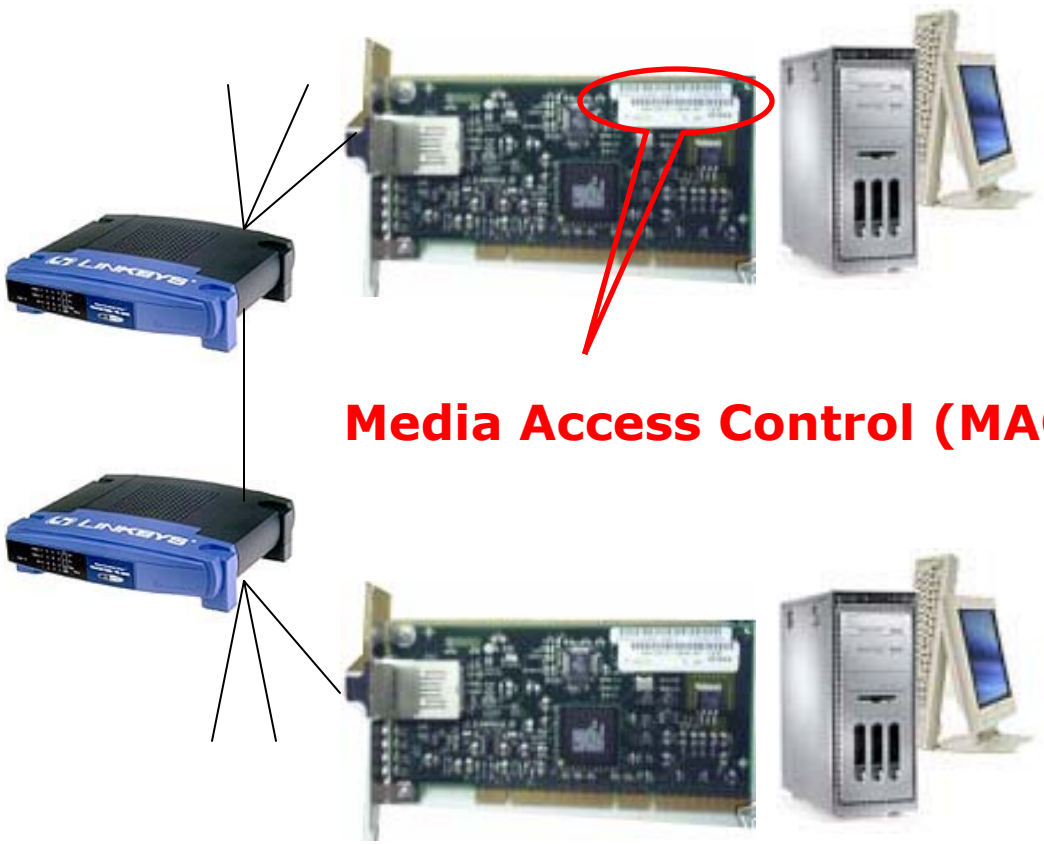
Network Layer:
[Ethernet][Token]

LAN / WAN



- **Application Layer: HTTP, Telnet, etc.**
- **Transport Layer: [TCP] [UDP]**
- **Internet Layer: [IP] [ICMP]**
- **Link Layer: [ARP] [RARP]**
- **Network Layer: [Ethernet][Token]**

LAN / WAN



Media Access Control (MAC)

Application Layer:
HTTP, Telnet, etc.

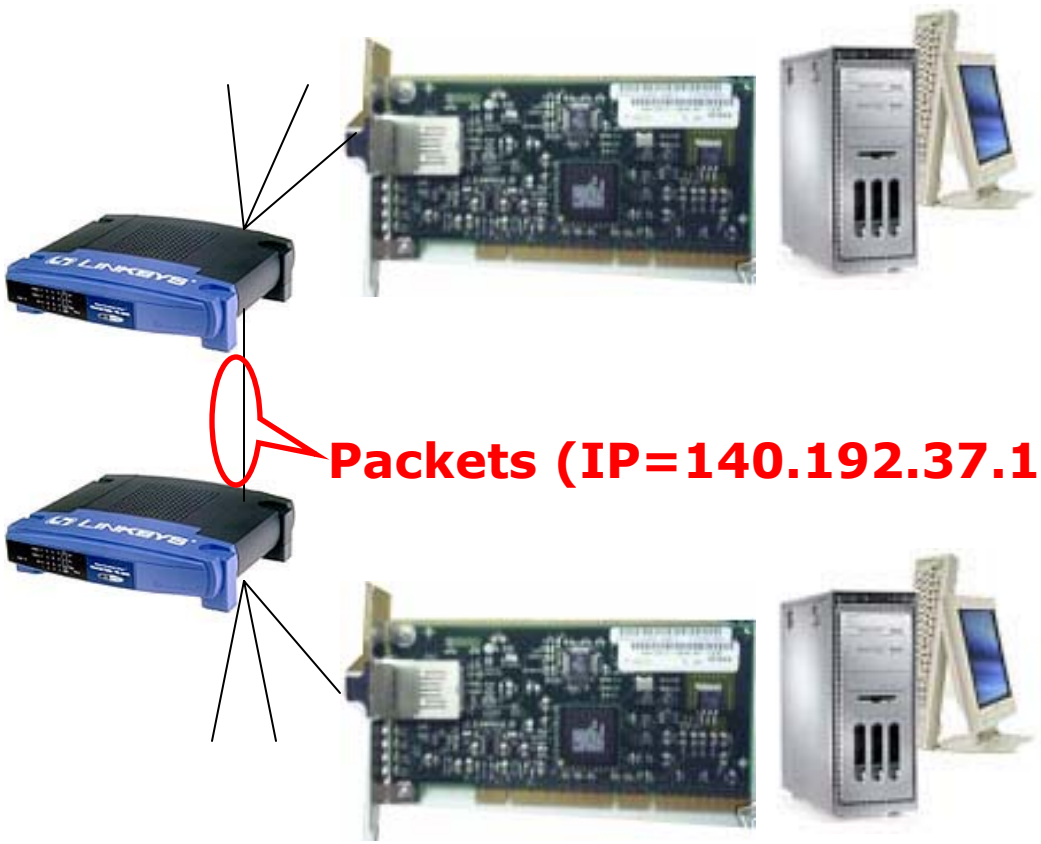
Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

LAN / WAN



Application Layer:
HTTP, Telnet, etc.

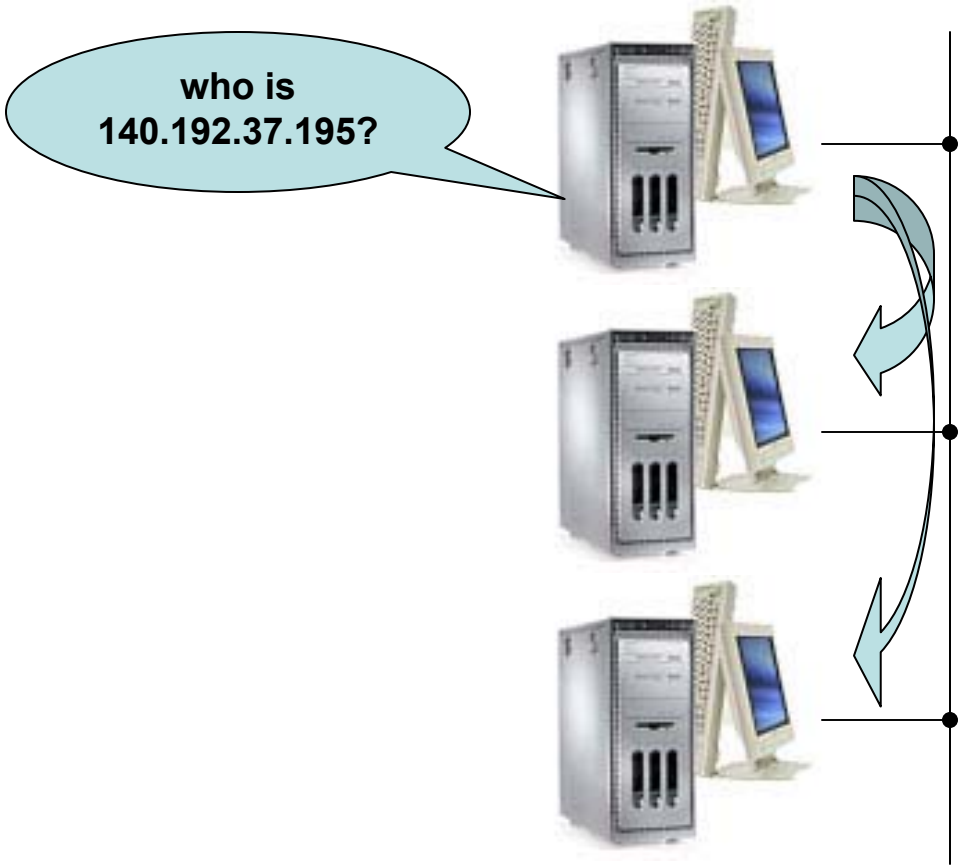
Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

LAN / ARP



Application Layer:
HTTP, Telnet, etc.

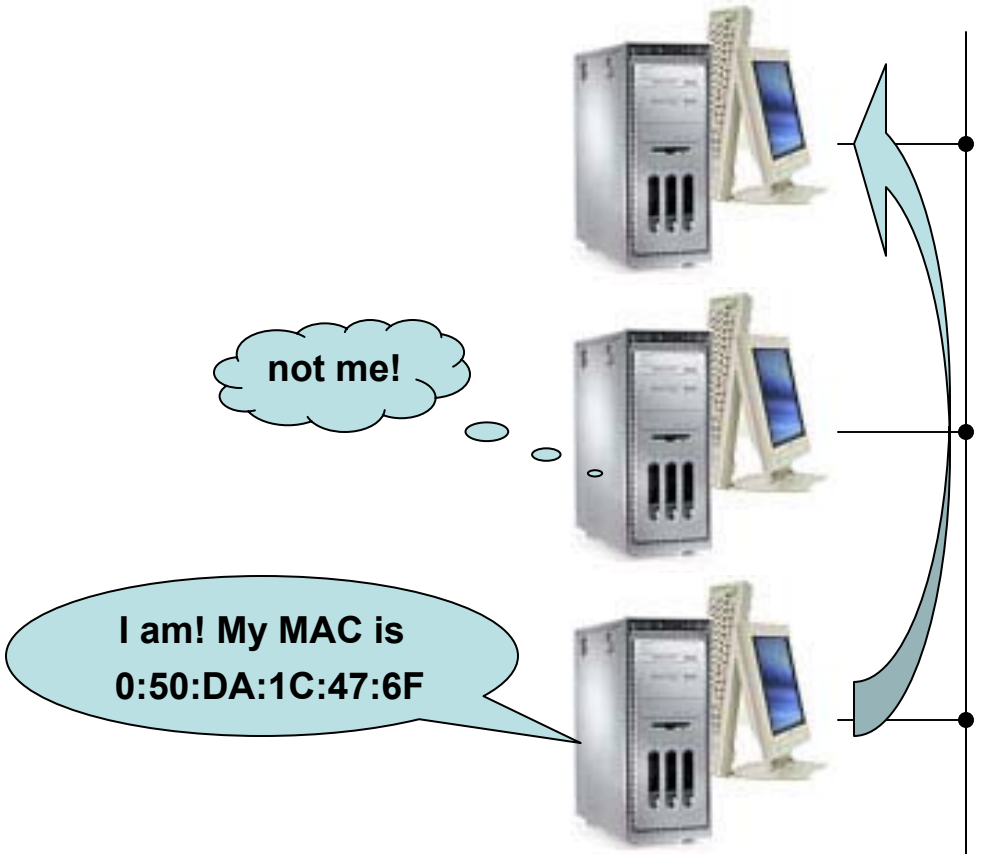
Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

LAN / ARP



Application Layer:
HTTP, Telnet, etc.

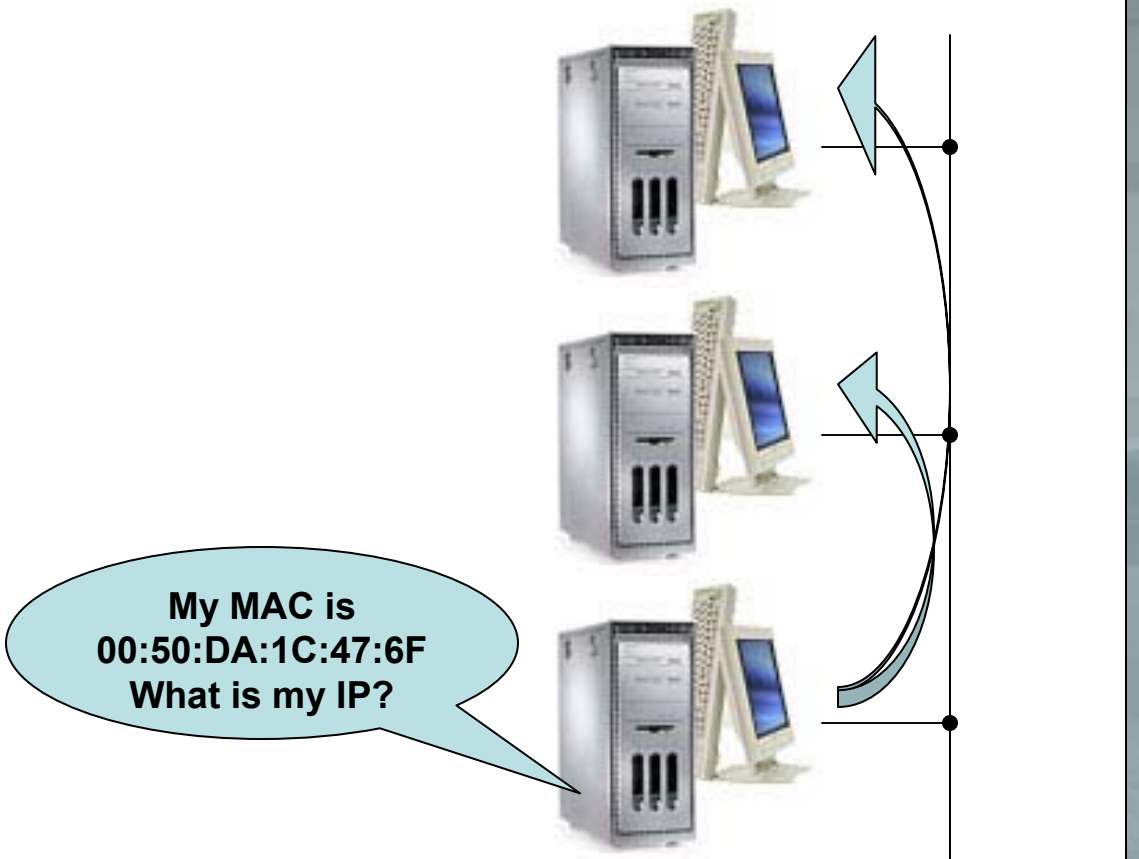
Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

LAN / RARP



Application Layer:
HTTP, Telnet, etc.

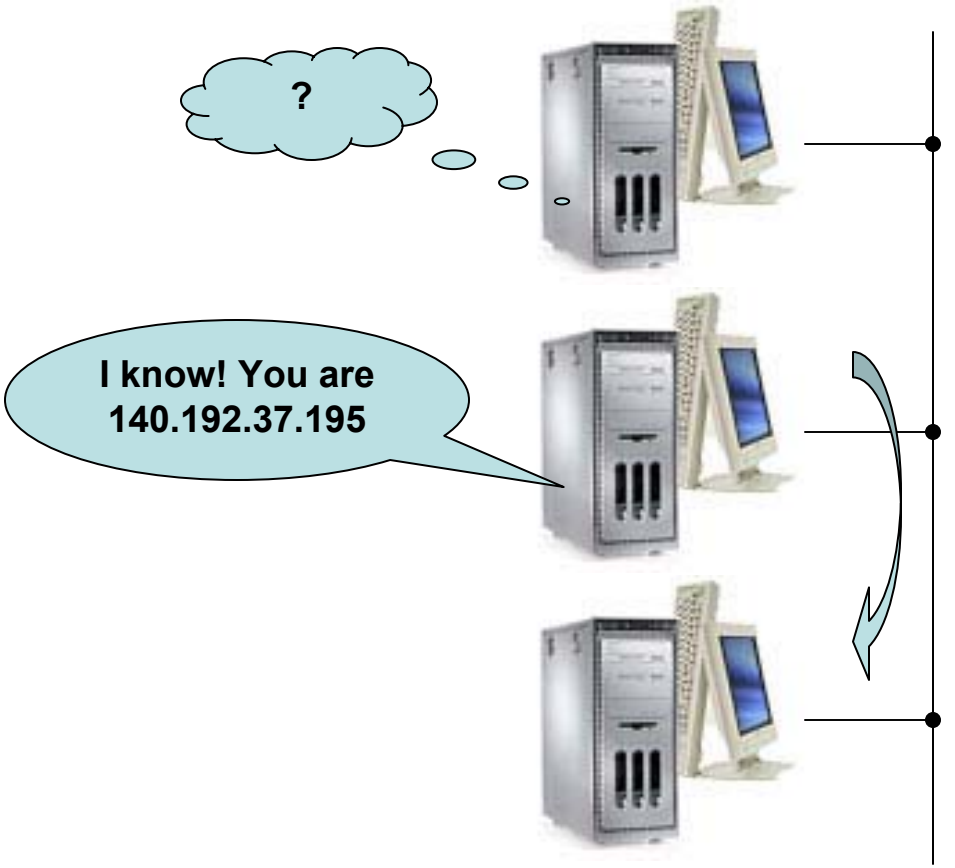
Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

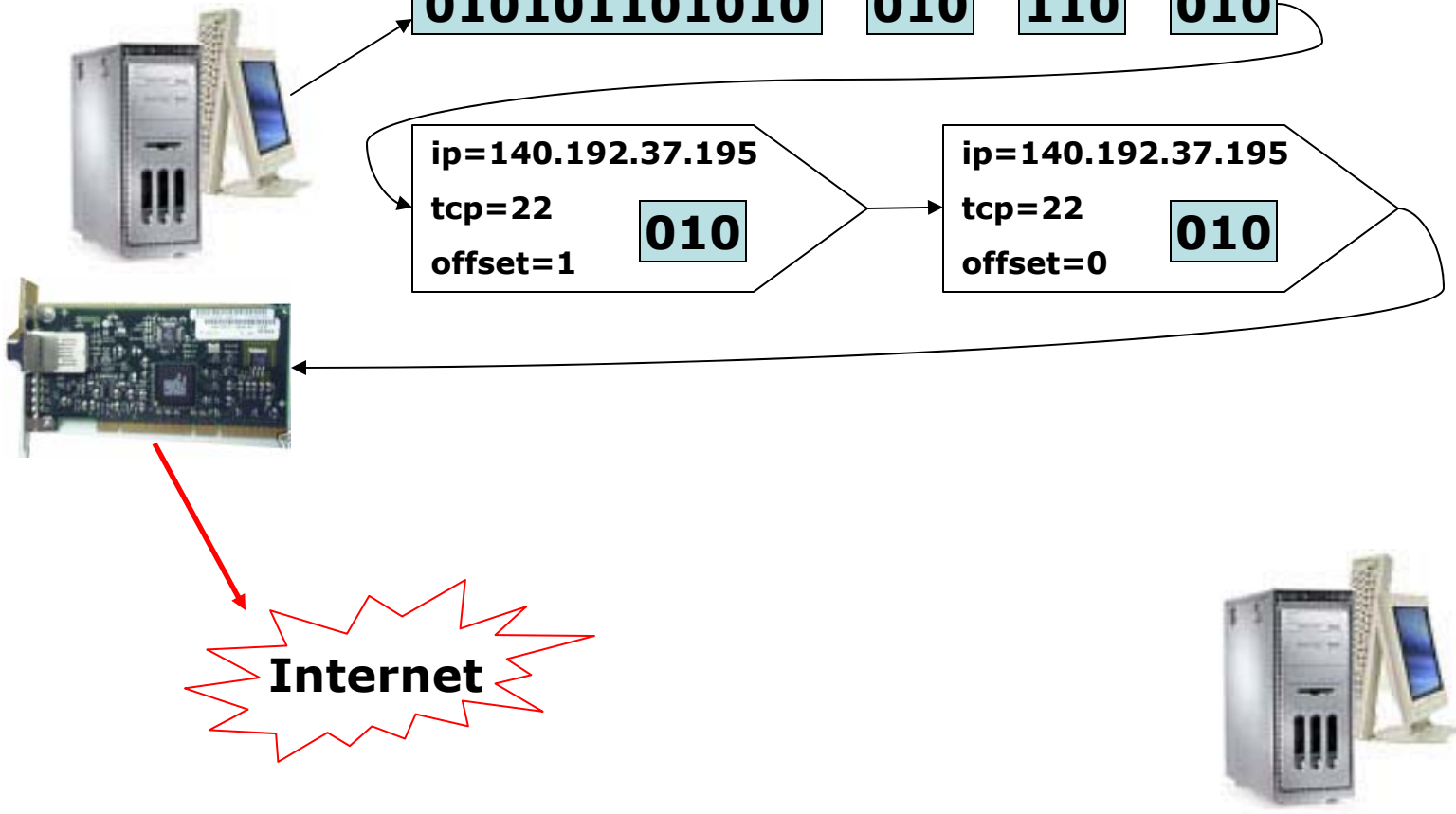
Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

LAN / RARP



send "010011010010...." to 140.192.37.195:22

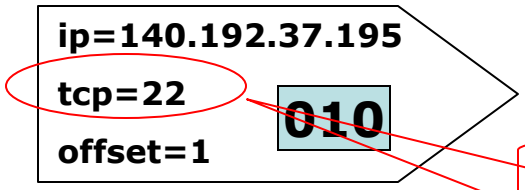


send "010011010010…." to 140.192.37.195:22

ip=140.192.37.195
tcp=22
offset=1
010

IP address of destination

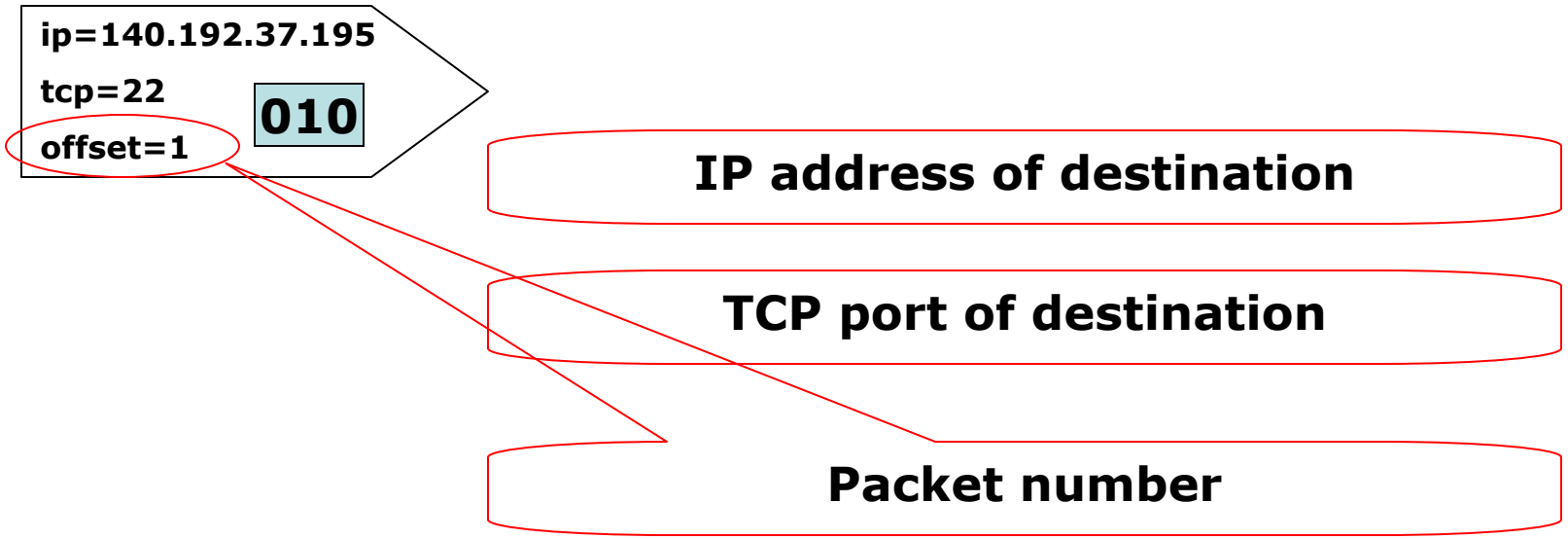
send "010011010010…." to 140.192.37.195:22



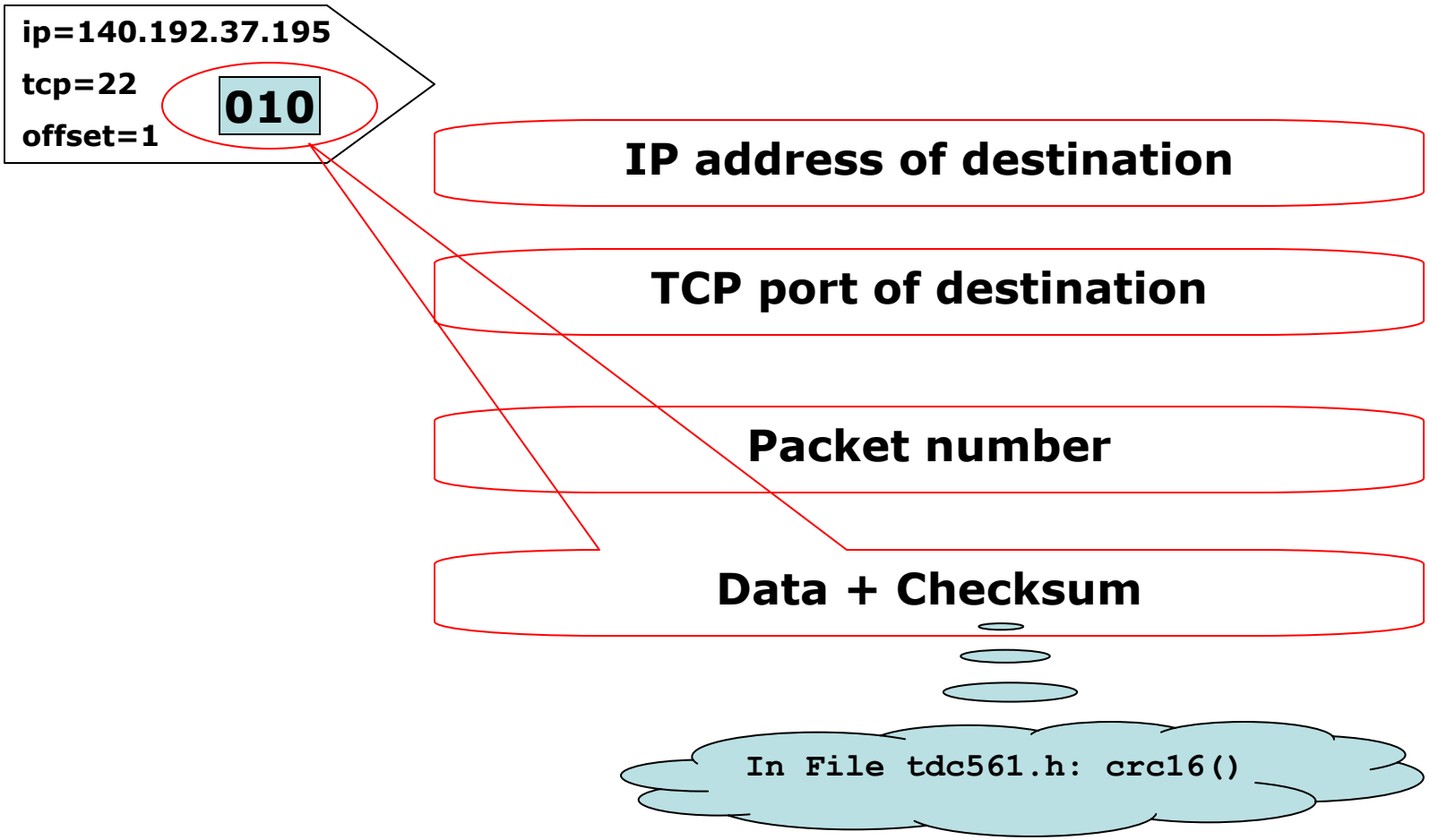
IP address of destination

TCP port of destination

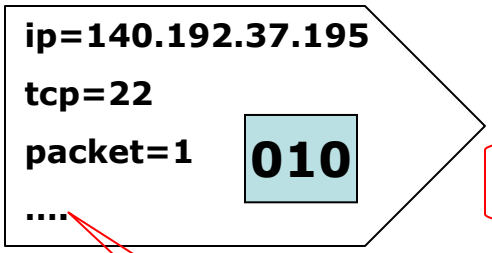
send "010011010010…." to 140.192.37.195:22



send "010011010010...." to 140.192.37.195:22



send "010011010010...." to 140.192.37.195:22



IP address of destination

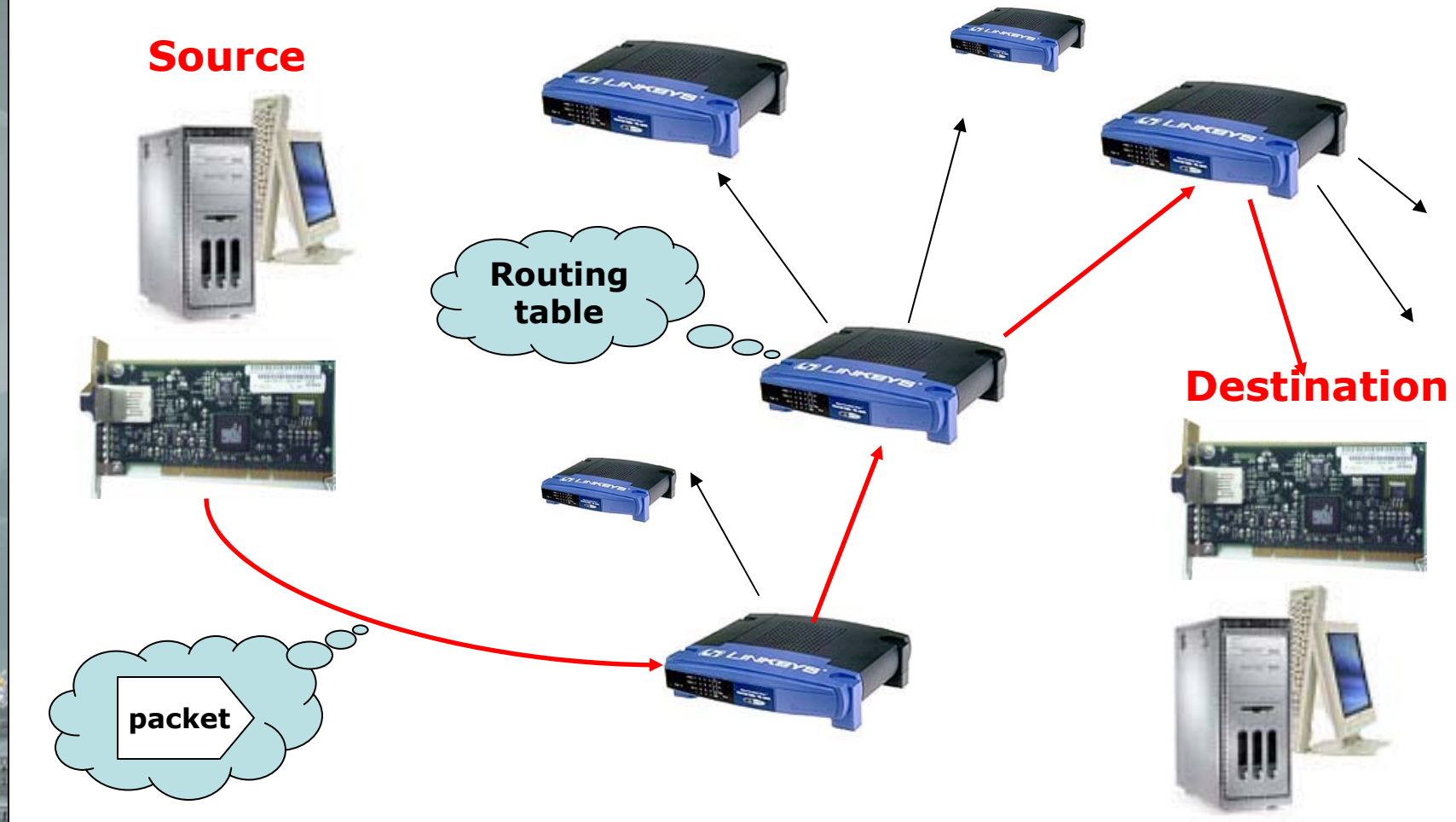
TCP port of destination

Packet number

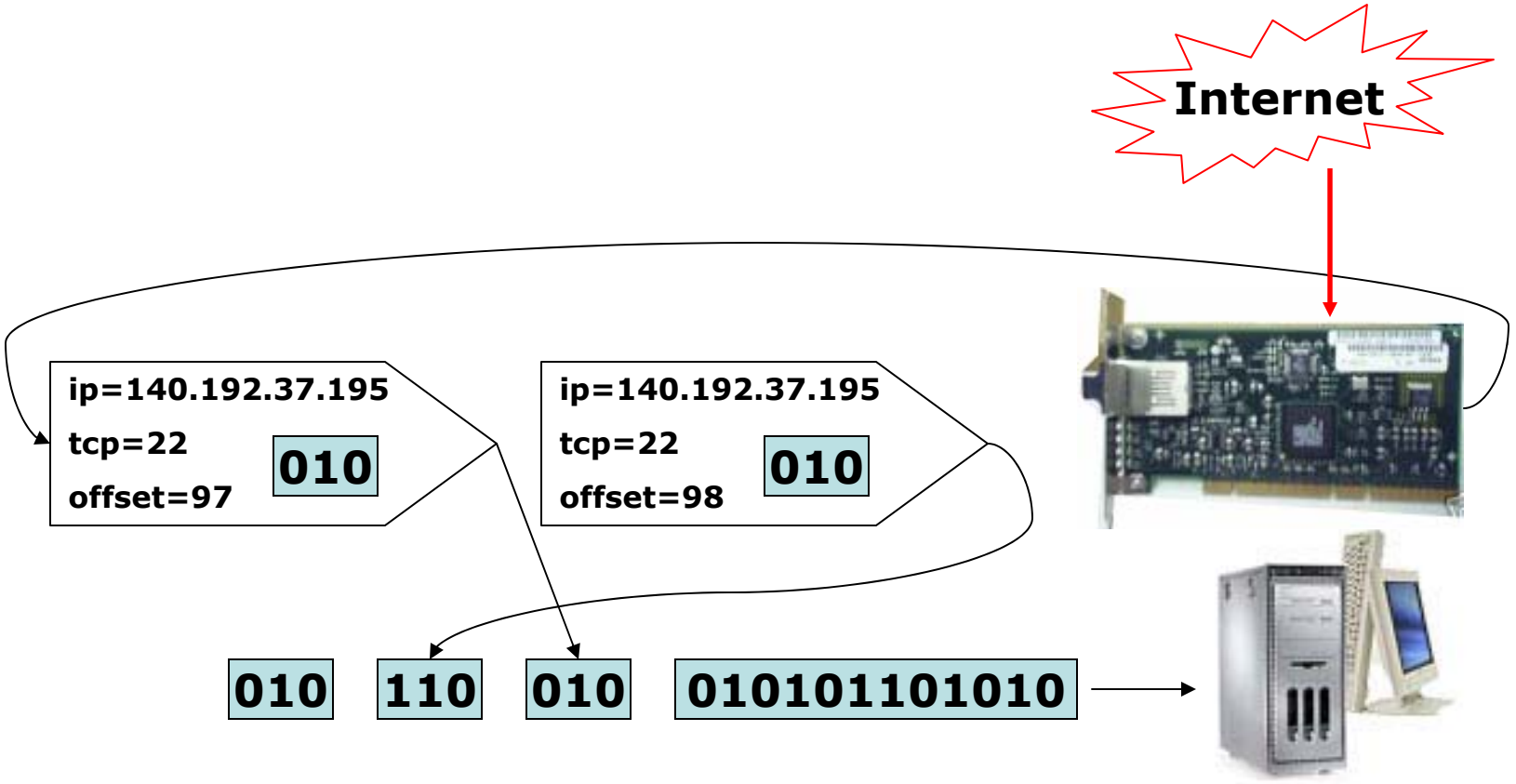
Data + Checksum

IP address and TCP port of source

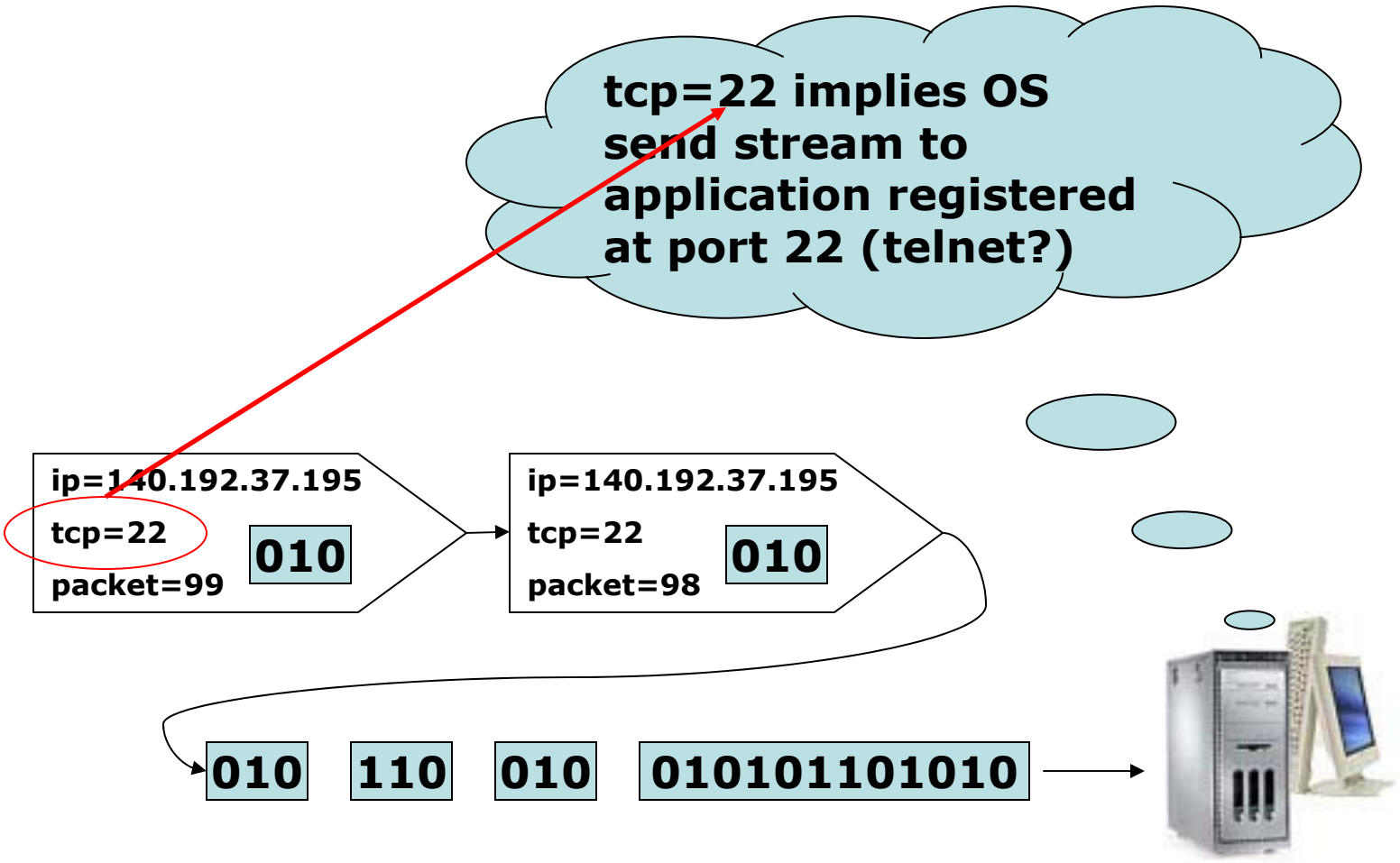
send "..." to 140.192.37.195:22



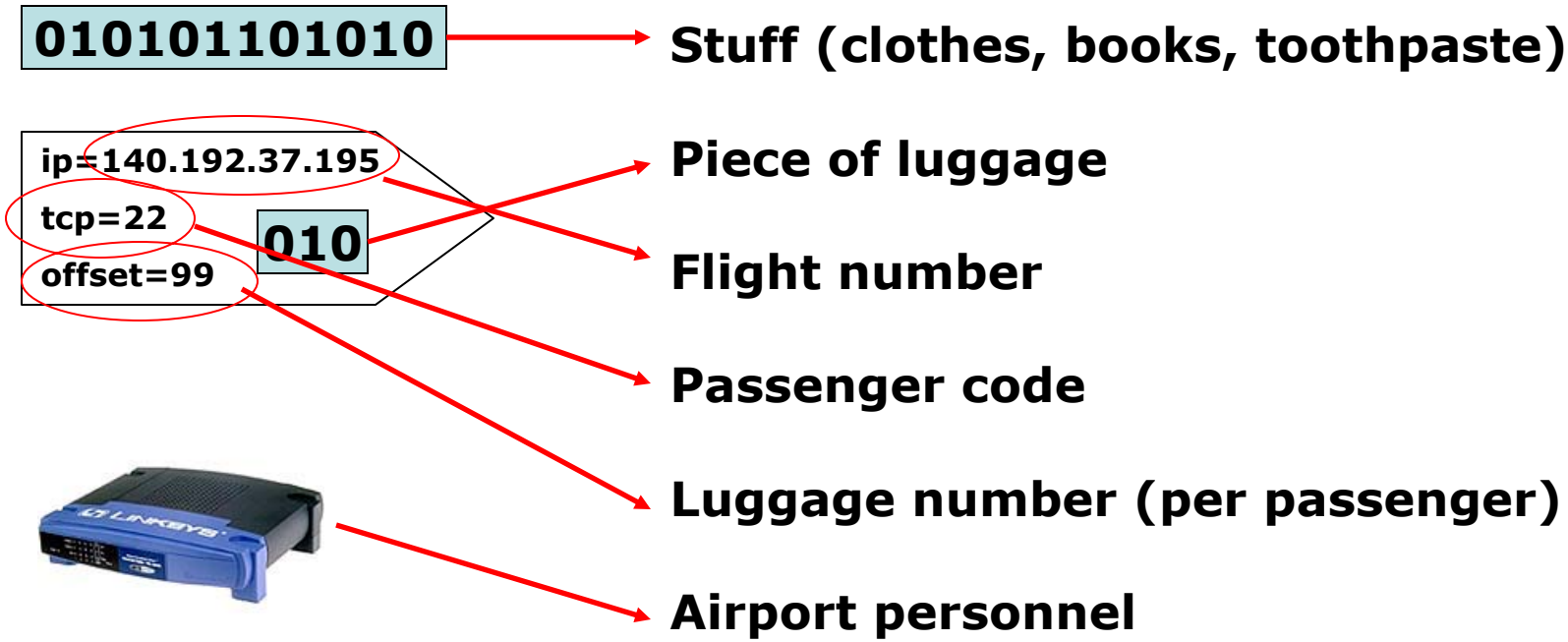
send "..." to 140.192.37.195:22



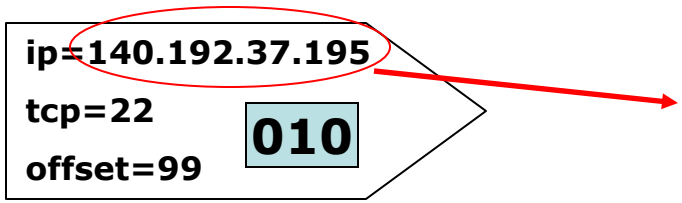
send "..." to 140.192.37.195:22



Airport analogy



IP format address



aaa.bbb.ccc.ddd

each field is 8 bits

entire address is 32 bits

Addresss types:

```
0xxxxxxx yyyyyyyy yyyyyyyy yyyyyyyy
10xxxxxx xxxxxxxx yyyyyyyy yyyyyyyy
110xxxxx xxxxxxxx xxxxxxxx yyyyyyyy
1110zzzz zzzzzzzz zzzzzzzz zzzzzzzz
```

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

http://www.netfor2.com/rfc791.txt

IP Packet Header Format (RFC 791)

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
...
```

Referece

- **Application Layer:** HTTP, Telnet, etc.
- **Transport Layer:** [TCP] [UDP]
- **Internet Layer:** [IP] [ICMP]
- **Link Layer:** [ARP] [RARP]
- **Network Layer:** [Ethernet][Token]

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
...
```

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
```

```
Bits      0-2: Precedence.
Bit 3:    0 = Normal Delay, 1 = Low Delay.
Bits 4:   0 = Normal Throughput, 1 = High Throughput.
Bits 5:   0 = Normal Relibility, 1 = High Relibility.
Bit 6-7: Reserved for Future Use.
```

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
...
```

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
...
```

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
...
```

```
Bit 0: reserved, must be zero
Bit 1: (DF) 0 = May Fragment, 1 = Don't Fragment.
Bit 2: (MF) 0 = Last Fragment, 1 = More Fragments.
```

- **Application Layer: HTTP, Telnet, etc.**
- **Transport Layer: [TCP] [UDP]**
- **Internet Layer: [IP] [ICMP]**
- **Link Layer: [ARP] [RARP]**
- **Network Layer: [Ethernet][Token]**

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
```

- **Application Layer: HTTP, Telnet, etc.**
- **Transport Layer: [TCP] [UDP]**
- **Internet Layer: [IP] [ICMP]**
- **Link Layer: [ARP] [RARP]**
- **Network Layer: [Ethernet][Token]**

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
```

- **Application Layer:** HTTP, Telnet, etc.
- **Transport Layer:** [TCP] [UDP]
- **Internet Layer:** [IP] [ICMP]
- **Link Layer:** [ARP] [RARP]
- **Network Layer:** [Ethernet][Token]

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
...
```

- **Application Layer:** HTTP, Telnet, etc.
- **Transport Layer:** [TCP] [UDP]
- **Internet Layer:** [IP] [ICMP]
- **Link Layer:** [ARP] [RARP]
- **Network Layer:** [Ethernet][Token]

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
...
```

- **Application Layer: HTTP, Telnet, etc.**
- **Transport Layer: [TCP] [UDP]**
- **Internet Layer: [IP] [ICMP]**
- **Link Layer: [ARP] [RARP]**
- **Network Layer: [Ethernet][Token]**

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
...
```

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

IP Packet Header Format

```
OCTET 1 Version (4 bit)+IHL (4 bit) (VER, IHL)
OCTET 2 Type of service (TOS)
OCTET 3,4 Total Length (TOL)
OCTET 5,6 Identification (ID)
OCTET 7,8 Flags (3 bit) + Offset (13 bit) (FLG, FRO)
OCTET 9 Time to Live (TTL)
OCTET 10 Protocol (PRO)
OCTET 11,12 Header Checksum (IP_SUM)
OCTET 13,14,15,16 Source Address (SRC)
OCTET 17,18,19,20 Destination Address (DEST)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25, 26 … Data

45 00 00 40 00 01 00 00 3C 11
E0 31 CE D9 8F 1F C7 B6 78 CB
...
```

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

TCP Header Format (RFC 793)

```
OCTET 1,2 Source Port (SRC_PORT)
OCTET 3,4 Destination Port (DEST_PORT)
OCTET 5,6,7,8 Sequence Number (SEQ)
OCTET 9,10,11,12 Acknowledgement Number (ACK)
OCTET 13,14 Offset(4 bit)+Reserved(6)+Control(6) (DTO,FLG)
OCTET 15,16 Window (WIN)
OCTET 17,18 Checksum (TCP_SUM)
OCTET 19,20 Urgent Pointer (URP)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25,26… Data
```

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

UDP Header Format (RFC 678)

```
OCTET 1,2 Source Port
OCTET 3,4 Destination Port
OCTET 5,6 Length
OCTET 7,8 Checksum
OCTET 9,10….. Data
```

Referece

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

TCP Header Format (RFC 793)

```
OCTET 1,2 Source Port (SRC_PORT)
OCTET 3,4 Destination Port (DEST_PORT)
OCTET 5,6,7,8 Sequence Number (SEQ)
OCTET 9,10,11,12 Acknowledgement Number (ACK)
OCTET 13,14 Offset(4 bit)+Reserved(6)+Control(6) (DTO,FLG)
OCTET 15,16 Window (WIN)
OCTET 17,18 Checksum (TCP_SUM)
OCTET 19,20 Urgent Pointer (URP)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25,26… Data
```

The sequence number of the first data octet in this segment (except when SYN is present). If SYN is present the sequence number is the initial sequence number (ISN) and the first data octet is ISN+1.

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

TCP Header Format (RFC 793)

```
OCTET 1,2 Source Port (SRC_PORT)
OCTET 3,4 Destination Port (DEST_PORT)
OCTET 5,6,7,8 Sequence Number (SEQ)
OCTET 9,10,11,12 Acknowledgement Number (ACK)
OCTET 13,14 Offset(4 bit)+Reserved(6)+Control(6) (DTO,FLG)
OCTET 15,16 Window (WIN)
OCTET 17,18 Checksum (TCP_SUM)
OCTET 19,20 Urgent Pointer (URP)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25,26… Data
```

If the ACK control bit is set this field contains the value of the next sequence number the sender of the segment is expecting to receive. Once a connection is established this is always sent.

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

TCP Header Format (RFC 793)

```
OCTET 1,2 Source Port (SRC_PORT)
OCTET 3,4 Destination Port (DEST_PORT)
OCTET 5,6,7,8 Sequence Number (SEQ)
OCTET 9,10,11,12 Acknowledgement Number (ACK)
OCTET 13,14 Offset(4 bit)+Reserved(6)+Control(6) (DTO,FLG)
OCTET 15,16 Window (WIN)
OCTET 17,18 Checksum (TCP_SUM)
OCTET 19,20 Urgent Pointer (URP)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25,26… Data
```

The number of 32 bit words in the TCP Header. This indicates where the data begins. The TCP header (even one including options) is an integral number of 32 bits long.

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

TCP Header Format (RFC 793)

```
OCTET 1,2 Source Port (SRC_PORT)
OCTET 3,4 Destination Port (DEST_PORT)
OCTET 5,6,7,8 Sequence Number (SEQ)
OCTET 9,10,11,12 Acknowledgement Number (ACK)
OCTET 13,14 Offset(4 bit)+Reserved(6)+Control(6) (DTO,FLG)
OCTET 15,16 Window (WIN)
OCTET 17,18 Checksum (TCP_SUM)
OCTET 19,20 Urgent Pointer (URP)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25,26… Data
```

URG: Urgent Pointer field significant
ACK: Acknowledgment field significant
PSH: Push Function
RST: Reset the connection
SYN: Synchronize sequence numbers
FIN: No more data from sender

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

TCP Header Format (RFC 793)

```
OCTET 1,2 Source Port (SRC_PORT)
OCTET 3,4 Destination Port (DEST_PORT)
OCTET 5,6,7,8 Sequence Number (SEQ)
OCTET 9,10,11,12 Acknowledgement Number (ACK)
OCTET 13,14 Offset(4 bit)+Reserved(6)+Control(6) (DTO,FLG)
OCTET 15,16 Window (WIN)
OCTET 17,18 Checksum (TCP_SUM)
OCTET 19,20 Urgent Pointer (URP)
OCTET 21,22,23 Options (OPT)
OCTET 24 Padding
OCTET 25,26… Data
```

The number of data octets beginning with the one indicated in the acknowledgment field which the sender of this segment is willing to accept.

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

struct sockaddr

```
#include <sys/socket.h>

struct sockaddr {
  unsigned short int sa_family;
  unsigned char sa_data[14];
};

sa_family = AF_INET   (ipv4)
          = AF_INET6  (ipv6)
          = AF_LOCAL / AF_UNIX (unix socket)
          = ...

sa_data   = ...
```

Referece

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

struct in_addr

```
#include <netinet/in.h>

struct in_addr {
  uint32_t s_addr;
};
```

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

struct sockaddr_in

```
#include <netinet/in.h>

struct sockaddr_in {
  short int sin_family;
  unsigned short int sin_port;
  struct in_addr sin_addr;
  char sin_zero[8];
};
```

Referece

- **Application Layer: HTTP, Telnet, etc.**
- **Transport Layer: [TCP] [UDP]**
- **Internet Layer: [IP] [ICMP]**
- **Link Layer: [ARP] [RARP]**
- **Network Layer: [Ethernet][Token]**

struct hostent

```
struct hostent {
  char *h_name; /* official name of host */
  char **h_aliases; /* alias list */
  int h_addrtype; /* host address type */
  int h_length; /* length of address */
  char **h_addr_list; /* list of addresses */
};

#define h_addr h_addr_list[0]
```

Referece

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

struct timeval

```
struct timeval {
  long int tv_sec
  long int tv_usec
};
```

Referece

Application Layer:
HTTP, Telnet, etc.

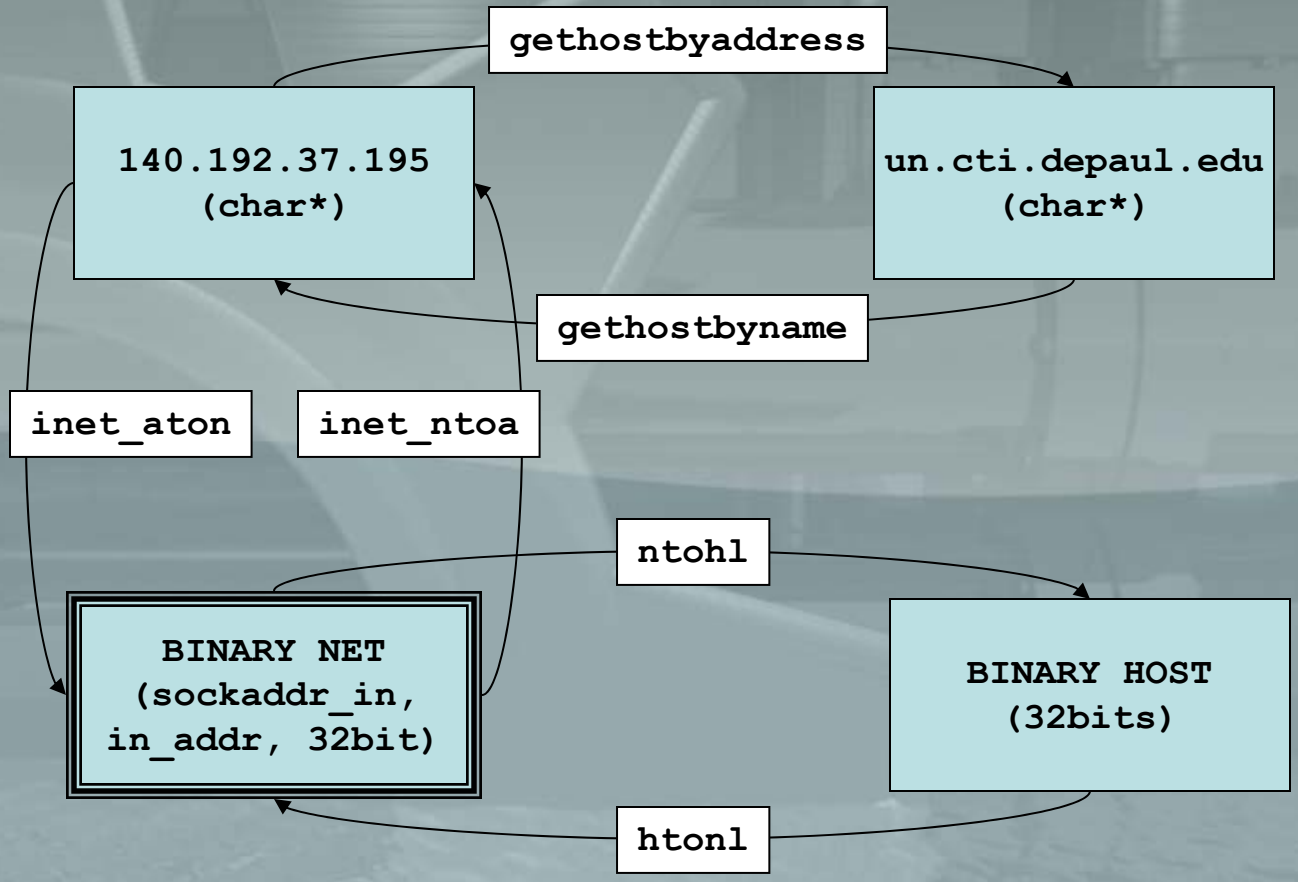
Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

Converting IP addresses



Referece

Application Layer:
HTTP, Telnet, etc.

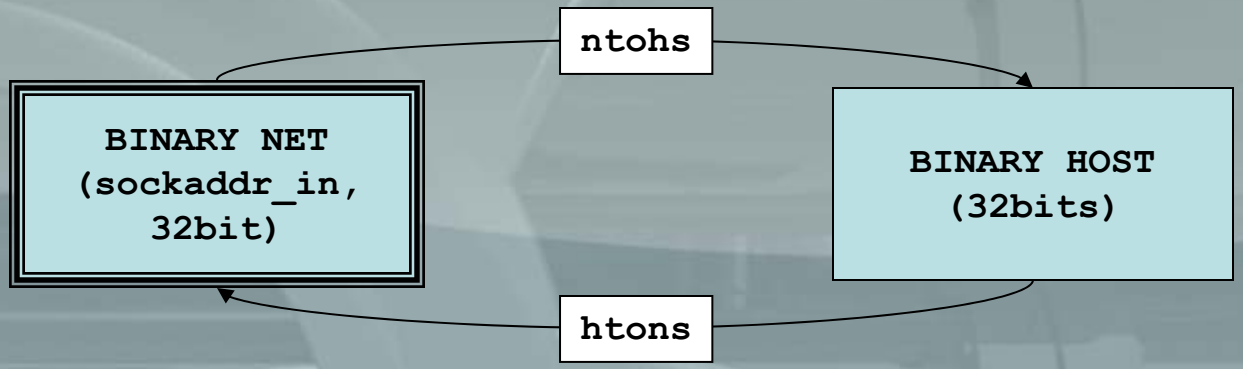
Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

Converting PORT number



Referece

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

TCP

```
Source:
1. create socket
2. connect to server
3. send
4. recv
5. close connection
```

```
Destination:
1. create socket
2. bind socket to port
3. listen to port ...
4. accept client
5. recv
6. send
7. close connection
```

UDP (datagrams, connectionless)

```
Source:
1. create socket
2. sendto
3. readfrom
```

```
Destination:
1. create socket
2. recvfrom
3. Sendto
```

Referece

Application Layer:
HTTP, Telnet, etc.

Transport Layer:
[TCP] [UDP]

Internet Layer:
[IP] [ICMP]

Link Layer:
[ARP] [RARP]

Network Layer:
[Ethernet][Token]

TCP

```
Source:
1. create socket
2. connect to server
3. send
4. recv
5. close connection
```

```
Destination:
1. create socket
2. bind socket to port
3. listen to port ...
4. accept client
5. recv
6. send
7. close connection
```

UDP (datagrams, connectionless)

```
Source:
1. create socket
2. sendto
3. readfrom
```

```
Destination:
1. create socket
2. recvfrom
3. Sendto
```

Referece