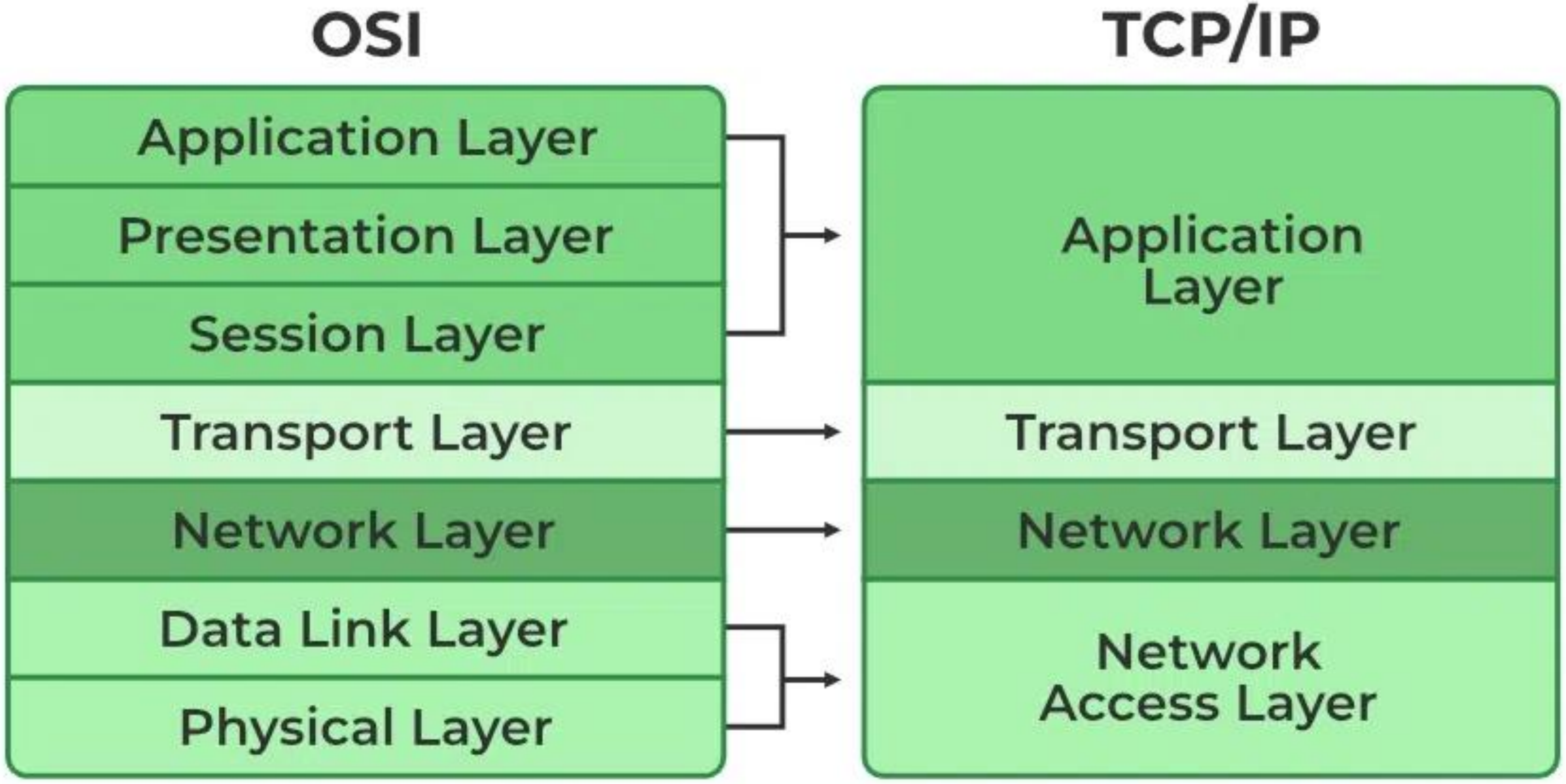
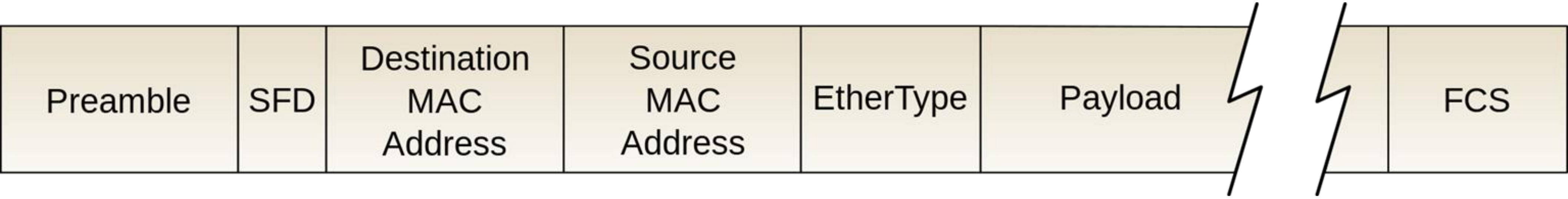
Cybersecurity
a.a. 2025-2026

Franco Arcieri


OSI
TCP/IP
Application Layer
Presentation Layer
Session Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer
Application Layer
Transport Layer
Network Layer
Network Access Layer

Struttura del pacchetto **ETHERNET**

- 6 bytes Destination Ethernet Address (Tutti 1 se broadcast, …)
- 6 bytes Source Ethernet Address
- 2 bytes Length or Type Field
- per IEEE 802.3 è il numero di bytes di dati
- per ethernet I o II è il "packet type", sempre > 1500(05DC)
- 46 bytes fino a 1500 sono dati! I pacchetti troppo corti devono essere riempiti fino ad almento 46 bytes
- 4 bytes (Frame Check sequence)



Network protocols

- Descrizioni dei tipi Ethernet
- 0x0000 0x05DC IEEE 802.3 Length Fields
- 0x0600 0x0600 Xerox XNS IDP
- 0x0800 0x0800 DOD IP
- 0x0801 0x0801 X.75 Internet
- 0x0802 0x0802 NBS Internet
- 0x0803 0x0803 ECMA Internet
- 0x0804 0x0804 CHAOSnet
- 0x0805 0x0805 X.25 Level 3
- 0x0806 0x0806 ARP (for IP and CHAOS)
- 0x0807 0x0807 Xerox XNS Compatibility
- 0x081C 0x081C Symbolics Private
- 0x0888 0x088A Xyplex
- 0x0900 0x0900 Ungermann-Bass network debugger
- 0x0A00 0x0A00 Xerox 802.3 PUP
- 0x0A01 0x0A01 Xerox 802.3 PUP Address Translation
- 0x0A02 0x0A02 Xerox PUP CAL Protocol (unused)
- 0x0BAD 0x0BAD Banyan Systems, Inc.
- 0x1000 0x1000 Berkeley Trailer negotiation

I protocolli incapsulati in ethernet

- 0x0000, 0x05DC, IEEE 802.3 Length Fields
- 0x0600, 0x0600, Xerox XNS IDP
- 0x0800, 0x0800, DOD IP
- 0x0801, 0x0801, X.75 Internet
- 0x0802, 0x0802, NBS Internet
- 0x0803, 0x0803, ECMA Internet
- 0x0804, 0x0804, CHAOSnet
- 0x0805, 0x0805, X.25 Level 3
- 0x0806, 0x0806, ARP (for IP and CHAOS)
- 0x0807, 0x0807, Xerox XNS Compatibility
- 0x081C, 0x081C, Symbolics Private
- 0x0888, 0x088A, Xyplex
- 0x0900, 0x0900, Ungermann-Bass network debugger
- 0x0A00, 0x0A00, Xerox 802.3 PUP
- 0x0A01, 0x0A01, Xerox 802.3 PUP Address Translation
- 0x0A02, 0x0A02, Xerox PUP CAL Protocol (unused)
- 0x0BAD, 0x0BAD, Banyan Systems, Inc.
- 0x1000, 0x1000, Berkeley Trailer negotiation
- 0x1001, 0x100F, Berkeley Trailer encapsulation for IP
- 0x1066, 0x1066, VALIS Systems
- 0x1600, 0x1600, VALID Systems
- 0x3C01, 0x3C0D, 3Com Corporation
- 0x3C10, 0x3C14, 3Com Corporation
- 0x4242, 0x4242, PCS Basic Block Protocol
- 0x5208, 0x5208, BBN Simnet Private
- 0x6000, 0x6000, DEC Unassigned
- 0x6001, 0x6001, DEC MOP Dump/Load Assistance
- 0x6002, 0x6002, DEC MOP Remote Console
- 0x6003, 0x6003, DEC DECnet Phase IV
- 0x6004, 0x6004, DEC LAT
- 0x6005, 0x6005, DEC DECnet Diagnostic Protocol: DECnet Customer Use
- 0x6007, 0x6007, DEC DECnet LAVC
- 0x6008, 0x6008, DEC Amber
- 0x6009, 0x6009, DEC MUMPS
- 0x6010, 0x6014, 3Com Corporation
- 0x7000, 0x7000, Ungermann-Bass download
- 0x7001, 0x7001, Ungermann-Bass NIU
- 0x7002, 0x7002, Ungermann-Bass diagnostic/loopback
- 0x7007, 0x7007, OS/9 Microware
- 0x7020, 0x7028, LRT (England)
- 0x7030, 0x7030, Proteon
- 0x7034, 0x7034, Cabletron
- 0x8003, 0x8003, Cronus VLN
- 0x8004, 0x8004, Cronus Direct
- 0x8005, 0x8005, HP Probe protocol
- 0x8006, 0x8006, Nestar
- 0x8008, 0x8008, AT&T
- 0x8010, 0x8010, Excelan
- 0x8013, 0x8013, SGI diagnostic type (obsolete)
- 0x8014, 0x8014, SGI network games (obsolete)
- 0x8015, 0x8015, SGI reserved type (obsolete)
- 0x8016, 0x8016, SGI bounce server (obsolete)
- 0x8019, 0x8019, Apollo
- 0x802E, 0x802E, Tymshare
- 0x802F, 0x802F, Tigan, Inc.
- 0x8035, 0x8035, Reverse ARP (RARP)
- 0x8036, 0x8036, Aeonic Systems
- 0x8038, 0x8038, DEC LANBridge
- 0x8039, 0x8039, DEC DSM
- 0x803A, 0x803A, DEC Aragon
- 0x803B, 0x803B, DEC VAXELN
- 0x803C, 0x803C, DEC NSMV
- 0x803D, 0x803D, DEC Ethernet CSMA/CD Encryption Protocol
- 0x803E, 0x803E, DEC DNA
- 0x803F, 0x803F, DEC LAN Traffic Monitor
- 0x8040, 0x8040, DEC NetBIOS
- 0x8041, 0x8041, DEC MS/DOS
- 0x8042, 0x8042, DEC Unassigned
- 0x8044, 0x8044, Planning Research Corporation
- 0x8046, 0x8046, AT&T
- 0x8047, 0x8047, AT&T
- 0x8049, 0x8049, ExperData (France)
- 0x805B, 0x805B, VMTP (Versatile Message Transaction Protocol, RFC-1045, Stanford)
- 0x805C, 0x805C, Stanford V Kernel production, Version 6.0
- 0x805D, 0x805D, Evans & Sutherland
- 0x8060, 0x8060, Little Machines
- 0x8062, 0x8062, Counterpoint Computers
- 0x8065, 0x8065, University of Massachusetts, Amherst
- 0x8066, 0x8066, University of Massachusetts, Amherst
- 0x8067, 0x8067, Veeco Integrated Automation
- 0x8068, 0x8068, General Dynamics
- 0x8069, 0x8069, AT&T
- 0x806A, 0x806A, Autophon (Switzerland)
- 0x806C, 0x806C, ComDesign
- 0x806D, 0x806D, Compugraphic Corporation
- 0x806E, 0x8077, Landmark Graphics Corporation
- 0x807A, 0x807A, Matra (France)
- 0x807B, 0x807B, Dansk Data Elektronic A/S (Denmark)
- 0x807C, 0x807C, Merit Intermodal
- 0x807D, 0x807D, VitaLink Communications
- 0x807E, 0x807E, VitaLink Communications
- 0x807F, 0x807F, VitaLink Communications
- 0x8080, 0x8080, VitaLink Communications bridge
- 0x8081, 0x8081, Counterpoint Computers
- 0x8082, 0x8082, Counterpoint Computers
- 0x8083, 0x8083, Counterpoint Computers
- 0x8088, 0x8088, Xyplex
- 0x8089, 0x8089, Xyplex
- 0x808A, 0x808A, Xyplex
- 0x809B, 0x809B, AppleTalk and Kinetics AppleTalk over Ethernet
- 0x809C, 0x809C, Datability
- 0x809D, 0x809D, Datability
- 0x809E, 0x809E, Datability
- 0x809F, 0x809F, Spider Systems, Ltd. (England)
- 0x80A3, 0x80A3, Nixdorf Computer (West Germany)
- 0x80A4, 0x80B3, Siemens Gammasonics, Inc.
- 0x80C0, 0x80C0, Digital Communication Associates
- 0x80C1, 0x80C1, Digital Communication Associates
- 0x80C2, 0x80C2, Digital Communication Associates
- 0x80C3, 0x80C3, Digital Communication Associates
- 0x80C6, 0x80C6, Pacer Software
- 0x80C7, 0x80C7, Applitek Corporation
- 0x80C8, 0x80CC, Intergraph Corporation
- 0x80CD, 0x80CD, Harris Corporation
- 0x80CE, 0x80CE, Harris Corporation
- 0x80CF, 0x80D2, Taylor Inst.
- 0x80D3, 0x80D3, Rosemount Corporation
- 0x80D4, 0x80D4, Rosemount Corporation
- 0x80D5, 0x80D5, IBM SNA Services over Ethernet
- 0x80DD, 0x80DD, Varian Associates
- 0x80DE, 0x80DE, Integrated Solutions TRFS (Transparent Remote File System)
- 0x80DF, 0x80DF, Integrated Solutions
- 0x80E0, 0x80E3, Allen-Bradley
- 0x80E4, 0x80F0, Datability
- 0x80F2, 0x80F2, Retix
- 0x80F3, 0x80F3, Kinetics, AppleTalk ARP (AARP)
- 0x80F4, 0x80F4, Kinetics
- 0x80F5, 0x80F5, Kinetics
- 0x80F7, 0x80F7, Apollo Computer
- 0x80FF, 0x8103, Wellfleet Communications
- 0x8107, 0x8107, Symbolics Private
- 0x8108, 0x8108, Symbolics Private
- 0x8109, 0x8109, Symbolics Private
- 0x8130, 0x8130, Waterloo Microsystems
- 0x8131, 0x8131, VG Laboratory Systems
- 0x8137, 0x8137, Novell (old) NetWare IPX
- 0x8138, 0x8138, Novell
- 0x8139, 0x813D, KTI
- 0x9000, 0x9000, Loopback (Configuration Test Protocol)
- 0x9001, 0x9001, Bridge Communications XNS Systems Management
- 0x9002, 0x9002, Bridge Communications TCP/IP Systems Management
- 0x9003, 0x9003, Bridge Communications
- 0xFF00, 0xFF00, BBN VITAL LANBridge cache wakeup

Il protocollo IP

Struttura dei pacchetti DOD IP (a partire dal byte 14 del pacchetto ETHERNET ovvero dal primo byte dei dati a livello ethernet)
Documenti di riferimento: RFC791, www.protocols.com

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Version
  - Version field indicates the format of the Internet header.
- IHL
  - Internet header length is the length of the Internet header in 32-bit words. Points to the beginning of the data. The minimum value for a correct header is 5.
- Type of service
  - Indicates the quality of service desired. Networks may offer service precedence, meaning that they accept traffic only above a certain precedence at times of high load. There is a three-way trade-off between low delay, high reliability and high throughput.
  - Bits 0-2: Precedence

111	Network control.
110	Internetwork control.
101	CRITIC/ECP.
100	Flash override.
011	Flash.
010	Immediate.
001	Priority.
000	Routine.

  - Bit 3: Delay

0	Normal delay.
1	Low delay.

  - Bit 4: Throughput

0	Normal throughput.
1	High throughput.

  - Bit 5: Reliability

0	Normal reliability.
1	High reliability.

  - Bits 6-7: Reserved for future use.
- Total length
  - Length of the datagram measured in bytes, including the Internet header and data. This field allows the length of a datagram to be up to 65,535 bytes, although such long datagrams are impractical for most hosts and networks. All hosts must be prepared to accept datagrams of up to 576 bytes, regardless of whether they arrive whole or in fragments. It is recommended that hosts send datagrams larger than 576 bytes only if the destination is prepared to accept the larger datagrams.
- Identification
  - Identifying value assigned by the sender to aid in assembling the fragments of a datagram.

- Flags
    - 3 bits. Control flags:
        - Bit 0 is reserved and must be zero.
        - Bit 1: Don’t fragment bit:
        - 0 May fragment.
        - 1 Don’t fragment.
        - Bit 2: More fragments bit:
        - 0 Last fragment.
        - 1 More fragments.
- Fragment offset
    - 13 bits. Indicates where this fragment belongs in the datagram. The fragment offset is measured in units of 8 bytes (64 bits). The first fragment has offset zero.
- Time to live
    - Indicates the maximum time the datagram is allowed to remain in the Internet system. If this field contains the value zero, the datagram must be destroyed. This field is modified in Internet header processing. The time is measured in units of seconds. However, since every module that processes a datagram must decrease the TTL by at least one (even if it processes the datagram in less than 1 second), the TTL must be thought of only as an upper limit on the time a datagram may exist. The intention is to cause undeliverable datagrams to be discarded and to bound the maximum datagram lifetime.
- Protocol
    - Indicates the next level protocol used in the data portion of the Internet datagram.
- Header checksum
    - A checksum on the header only. Since some header fields change, e.g., Time To Live, this is recomputed and verified at each point that the Internet header is processed.
- Source address / destination address
    - 32 bits each. A distinction is made between names, addresses and routes. A name indicates an object to be sought. An address indicates the location of the object. A route indicates how to arrive at the object. The Internet protocol deals primarily with addresses. It is the task of higher level protocols (such as host-to-host or application) to make the mapping from names to addresses. The Internet module maps Internet addresses to local net addresses. It is the task of lower level procedures (such as local net or gateways) to make the mapping from local net addresses to routes.

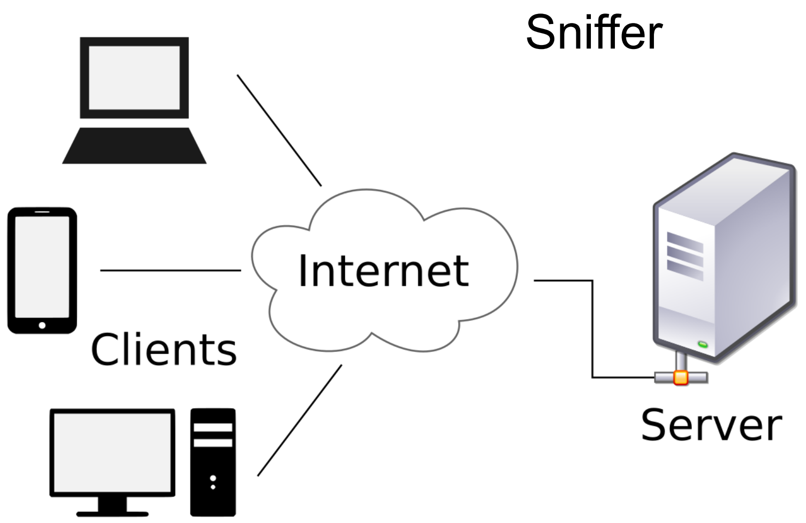
- Options
  - Options may or may not appear in datagrams. They must be implemented by all IP modules (host and gateways). What is optional is their transmission in any particular datagram, not their implementation. In some environments, the security option may be required in all datagrams.
- The option field is variable in length. There may be zero or more options. There are two possible formats for an option:
  - A single octet of option type.
  - An option type octet, an option length octet and the actual option data octets.
- The length octet includes the option type octet and the actual option data octets.
- The option type octet has 3 fields:
  - 1 bit: Copied flag. Indicates that this option is copied into all fragments during fragmentation:
    - 0 Copied.
    - 1 Not copied.
  - 2 bits: Option class
    - 0 Control.
    - 1 Reserved for future use.
    - 2 Debugging and measurement.
    - 3 Reserved for future use.
- 5 bits: Option number.
- Data
  - IP data or higher layer protocol header.

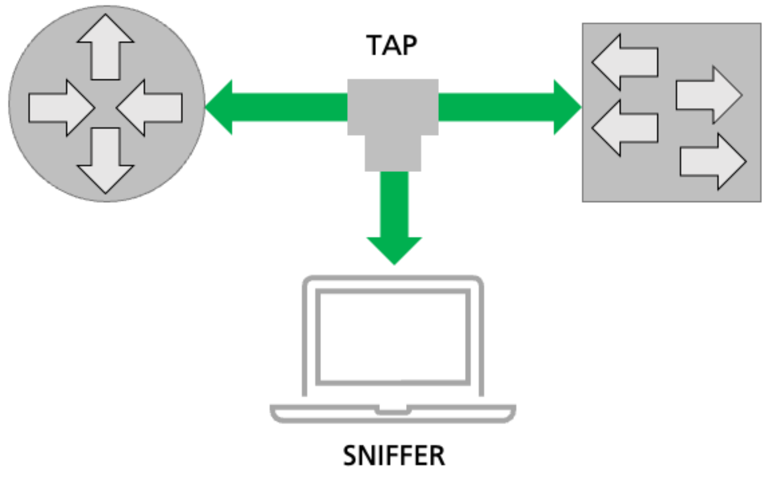
I protocolli incapsulati in IP

0, Reserved, Reserved
1, ICMP, Internet Control Message
2, IGMP, Internet Group Management
3, GGP, Gateway-to-Gateway
4, IP, IP in IP (encasulation)
5, ST, Stream
6, TCP, Transmission Control
7, UCL, UCL
8, EGP, Exterior Gateway Protocol
9, IGP, any private interior gateway
10, BBN-RCC-MON, BBN RCC Monitoring
11, NVP-II, Network Voice Protocol
12, PUP, PUP
13, ARGUS, ARGUS
14, EMCON, EMCON
15, XNET, Cross Net Debugger
16, CHAOS, Chaos
17, UDP, User Datagram
18, MUX, Multiplexing
19, DCN-MEAS, DCN Measurement Subsystems
20, HMP, Host Monitoring
21, PRM, Packet Radio Measurement
22, XNS-IDP, XEROX NS IDP
23, TRUNK-1, Trunk-1
24, TRUNK-2, Trunk-2
25, LEAF-1, Leaf-1
26, LEAF-2, Leaf-2
27, RDP, Reliable Data Protocol
28, IRTP, Internet Reliable Transaction
29, ISO-TP4, ISO Transport Protocol Class 4
30, NETBLT, Bulk Data Transfer Protocol
31, MFE-NSP, MFE Network Services Protocol
32, MERIT-INP, MERIT Internodal Protocol
33, SEP, Sequential Exchange Protocol
34, 3PC, Third Party Connect Protocol
35, IDPR, Inter-Domain Policy Routing Protocol
36, XTP, XTP
37, DDP, Datagram Delivery Protocol
38, IDPR-CMTP, IDPR Control Message Transport Protocol
39, TP++, TP++ Transport Protocol
40, IL, IL Transport Protocol
41, SIP, Simple Internet Protocol
42, SDRP, Source Demand Routing Protocol
43, SIP-SR, SIP Source Route
44, SIP-FRAG, SIP Fragment
45, IDRP, Inter-Domain Routing Protocol
46, RSVP, Reservation Protocol
47, GRE, General Routing Encapsulation
48, MHRP, Mobile Host Routing Protocol
49, BNA, BNA
50, SIPP-ESP, SIPP Encap Security Payload
51, SIPP-AH, SIPP Authentication Header
52, I-NLSP, Integrated Net Layer Security TUBA
53, SWIPE, IP with Encryption
54, NHRP, NBMA Next Hop Resolution Protocol
61, any host, any host internal protocol
62, CFTP, CFTP
63, any net, local network
64, SAT-EXPAK, SATNET and Backroom EXPAK
65, KRYPTOLAN, Kryptolan
66, RVD, MIT Remote Virtual Disk Protocol
67, IPPC, Internet Pluribus Packet Core
68, any file, distributed file system
69, SAT-MON, SATNET Monitoring
70, VISA, VISA Protocol
71, IPCV, Internet Packet Core Utility
72, CPNX, Computer Protocol Network Executive
73, CPHB, Computer Protocol Heart Beat
74, WSN, Wang Span Network
75, PVP, Packet Video Protocol
76, BR-SAT-MON, Backroom SATNET Monitoring
77, SUN-ND, SUN ND PROTOCOL-Temporary
78, WB-MON, WIDEBAND Monitoring
79, WB-EXPAK, WIDEBAND EXPAK
80, ISO-IP, ISO Internet Protocol
81, VMTP, VMTP
82, SECURE-VMTP, SECURE-VMTP
83, VINES, VINES
84, TTP, TTP
85, NSFNET-IGP, NSFNET-IGP
86, DGP, Dissimilar Gateway Protocol
87, TCF, TCF
88, IGRP, IGRP
89, OSPFIGP, OSPFIGP
90, Sprite-RPC, Sprite RPC Protocol
91, LARP, Locus Address Resolution Protocol
92, MTP, Multicast Transport Protocol
93, AX.25, AX.25 Frames
94, IPIP, IP-within-IP Encapsulation Protocol
95, MICP, Mobile Internetworking Control Pro.
96, SCC-SP, Semaphore Communications Sec. Pro.
97, ETHERIP, Ethernet-within-IP Encapsulation
98, ENCAP, Encapsulation Header
99, any scheme, any private encryption scheme
100, GMTP, GMTP
255, Reserved, Reserved

Sniffing della rete

- tcpdump, tool più famoso per fare sniffing
  - Wireshark il più semplice



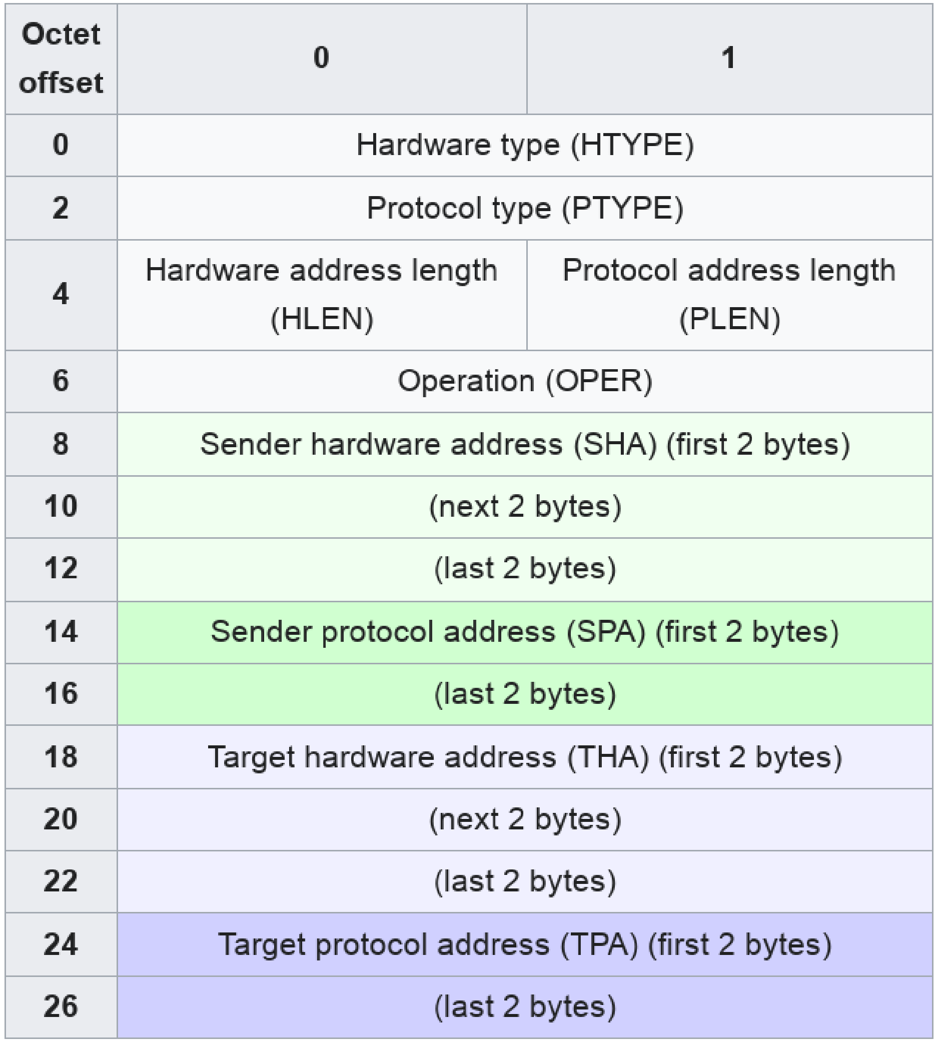


ARP

- The Address Resolution Protocol uses a simple message format containing one address resolution request or response. The packets are carried at the data link layer of the underlying network as raw payload. In the case of Ethernet, a 0x0806 EtherType value is used to identify ARP frames.
- The size of the ARP message depends on the link layer and network layer address sizes. The message header specifies the types of network in use at each layer as well as the size of addresses of each. The message header is completed with the operation code for request (1) and reply (2). The payload of the packet consists of four addresses, the hardware and protocol address of the sender and receiver hosts.
- The principal packet structure of ARP packets is shown in the following table which illustrates the case of IPv4 networks running on Ethernet. In this scenario, the packet has 48-bit fields for the sender hardware address (SHA) and target hardware address (THA), and 32-bit fields for the corresponding sender and target protocol addresses (SPA and TPA). The ARP packet size in this case is 28 bytes.

ARP

Internet Protocol (IPv4) over Ethernet ARP packet



Octet offset	0	1
0	Hardware type (HTYPE)	
2	Protocol type (PTYPE)	
4	Hardware address length (HLEN)	Protocol address length (PLEN)
6	Operation (OPER)	
8	Sender hardware address (SHA) (first 2 bytes)	
10	(next 2 bytes)	
12	(last 2 bytes)	
14	Sender protocol address (SPA) (first 2 bytes)	
16	(last 2 bytes)	
18	Target hardware address (THA) (first 2 bytes)	
20	(next 2 bytes)	
22	(last 2 bytes)	
24	Target protocol address (TPA) (first 2 bytes)	
26	(last 2 bytes)	

- Hardware type (HTYPE)
  - This field specifies the network link protocol type. Example: Ethernet is 1.[2]
- Protocol type (PTYPE)
  - This field specifies the internetwork protocol for which the ARP request is intended. For IPv4, this has the value 0x0800. The permitted PTYPE values share a numbering space with those for EtherType.[2][3]
- Hardware length (HLEN)
  - Length (in octets) of a hardware address. Ethernet address length is 6.
- Protocol length (PLEN)
  - Length (in octets) of internetwork addresses. The internetwork protocol is specified in PTYPE. Example: IPv4 address length is 4.
- Operation
  - Specifies the operation that the sender is performing: 1 for request, 2 for reply.
- Sender hardware address (SHA)
  - Media address of the sender. In an ARP request this field is used to indicate the address of the host sending the request. In an ARP reply this field is used to indicate the address of the host that the request was looking for.
- Sender protocol address (SPA)
  - Internetwork address of the sender.
- Target hardware address (THA)
  - Media address of the intended receiver. In an ARP request this field is ignored. In an ARP reply this field is used to indicate the address of the host that originated the ARP request.
- Target protocol address (TPA)
  - Internetwork address of the intended receiver.
- ARP protocol parameter values have been standardized and are maintained by the Internet Assigned Numbers Authority (IANA).[2]
- The EtherType for ARP is 0x0806. This appears in the Ethernet frame header when the payload is an ARP packet and is not to be confused with PTYPE, which appears within this encapsulated ARP packet.

ARP

- Con il protocollo ARP posso scoprire chi ha uno specifico indirizzo IP
- Ma come posso scoprire la topologia della mia rete?
  - In particolare quante schede di rete / device sono presenti nella mia rete locale?
- Ma come faccio a andare su internet, o meglio, come posso comunicare con un dispositivo che si trova su un’altra rete locale?
- Posso scoprire la topologia di una rete locale cui non appartengo?

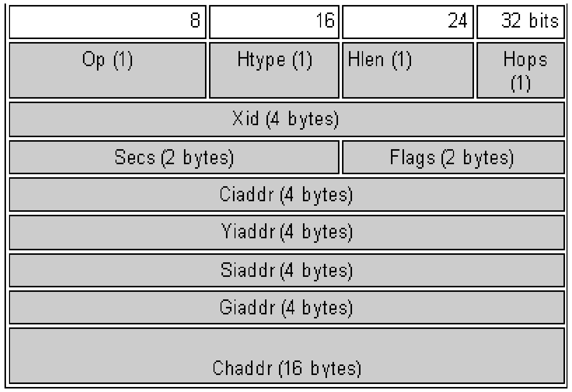
Cerchiamo gli indirizzi MAC della mia rete locale

- Conoscere il vostro MAC address
- Conoscere il vostro IP address
- Inviare una richiesta ARP a tutti gli indirizzi IP definiti tramite la maschera di sottorete (in genere 255.255.255.0 e quindi i 256 indirizzi IP che ricavate dal vostro IP sostituendo al byte meno significativo di valori da 0 a 255
- Per le richieste di cui avete risposta, allora conoscerete l’associazione MAC address, indirizzo IP

Struttura dei pacchetti DHCP (immediatamente successivi ai pacchetti DOD IP)

- Documenti di riferimento: RFC1531, www.protocols.com
- The Dynamic Host Configuration Protocol (DHCP) provides Internet hosts with configuration parameters. DHCP is an extension of BOOTP. DHCP consists of two components: a protocol for delivering host-specific configuration parameters from a DHCP server to a host and a mechanism for allocation of network addresses to hosts.

- DHCP header structure
  - Op
    - The message operation code. Messages can be either BOOTREQUEST or BOOTREPLY.
  - Htype
    - The hardware address type.
  - Hlen
    - The hardware address length.
  - Xid
    - The transaction ID.
  - Secs
    - The seconds elapsed since the client began the address acquisition or renewal process.
  - Flags
    - The flags.
  - Ciaddr
    - The client IP address.
  - Yiaddr
    - The "Your" (client) IP address.
  - Siaddr
    - The IP address of the next server to use in bootstrap.
  - Giaddr
    - The relay agent IP address used in booting via a relay agent.
  - Chaddr
    - The client hardware address.

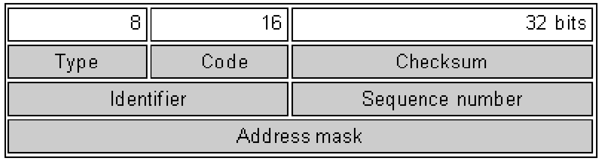


DHCP header structure

ICMP

- Struttura dei pacchetti ICMP (immediatamente successivi ai pacchetti IP)
- Documenti di riferimento: RFC792, www.protocols.com
- IETF RFC792 defines the Internet Control Message Protocol (ICMP). ICMP messages generally contain information about routing difficulties with IP datagrams or simple exchanges such as time-stamp or echo transactions.
- ICMP header structure

Type	Code	Description
0		Echo reply.
3		Destination unreachable.
3	*0*	*Net unreachable.*
3	*1*	*Host unreachable.*
3	*2*	*Protocol unreachable.*
3	*3*	*Port unreachable.*
3	*4*	*Fragmentation needed and DF set.*
3	*5*	Source route failed.
4		Source quench.
5		Redirect.
5	0	Redirect datagrams for the network.
5	1	Redirect datagrams for the host.
5	2	Redirect datagrams for the type of service and network.
5	3	Redirect datagrams for the type of service and host.
8		Echo.
11		Time exceeded.
11	0	Time to live exceeded in transit.
11	1	Fragment reassemble time exceeded.
12		Parameter problem.
13		Timestamp.
14		Timestamp reply.
15		Information request.
16		Information reply.



ICMP header structure

- Checksum
  - The 16-bit one’s complement of the one’s complement sum of the ICMP message starting with the ICMP Type. For computing the checksum, the checksum field should be zero.
- Identifier
  - An identifier to aid in matching requests/replies; may be zero.
- Sequence number
  - Sequence number to aid in matching requests/replies; may be zero.
- Address mask
  - A 32-bit mask.

UDP

- Struttura dei pacchetti UDP (immediatamente successivi ai pacchetti DOD IP)
- Documenti di riferimento: RFC768, www.protocols.com

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Length              |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             data                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

                         TCP Header Format
```

- The User Datagram Protocol (UDP), defined by IETF RFC768, provides a simple, but unreliable message service for transaction-oriented services. Each UDP header carries both a source port identifier and destination port identifier, allowing high-level protocols to target specific applications and services among hosts.

TCP

- Struttura dei pacchetti TCP (immediatamente successivi ai pacchetti DOD IP)
- Documenti di riferimento: RFC793, www.protocols.com

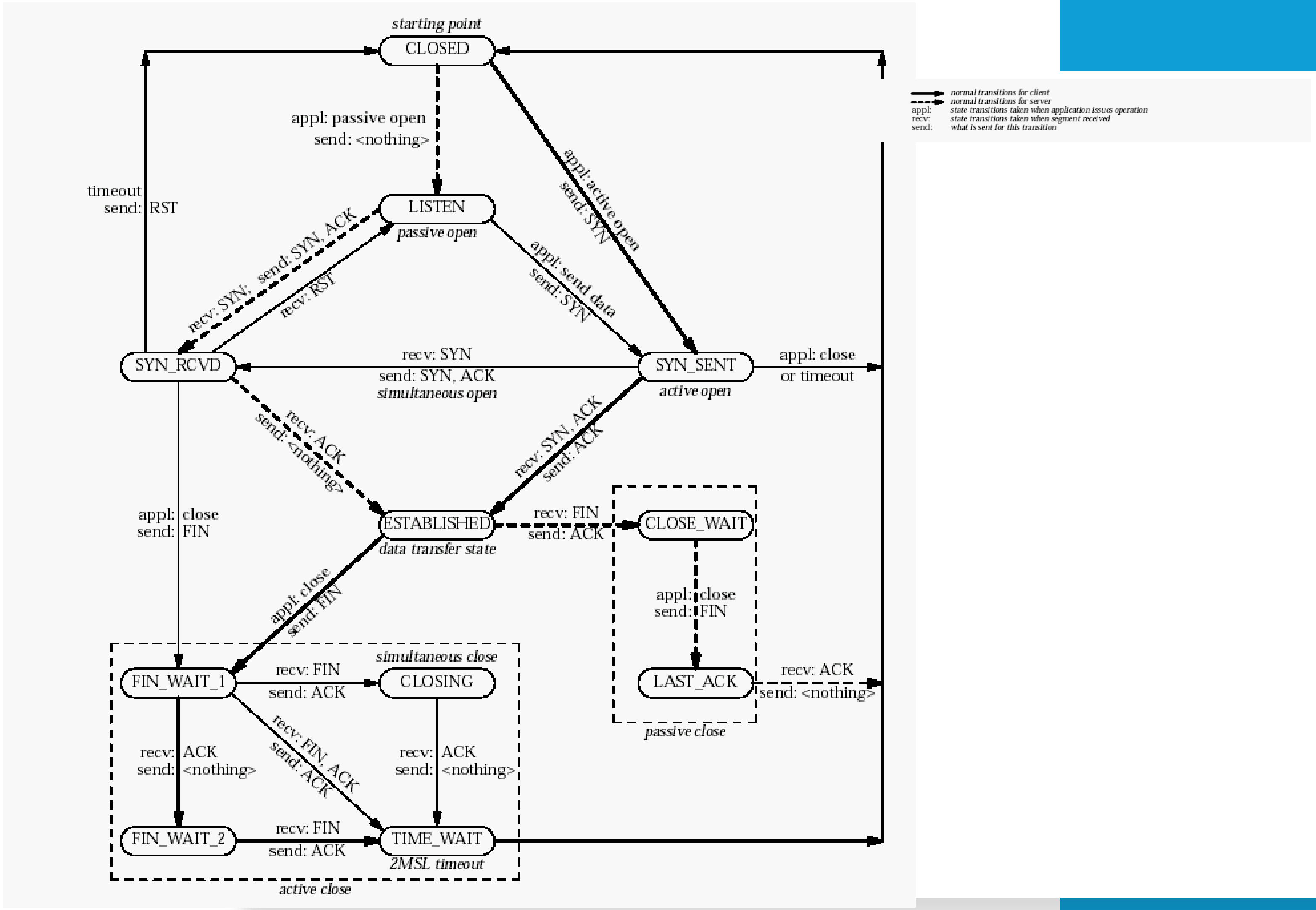
```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Source Port          |       Destination Port        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        Sequence Number                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    Acknowledgment Number                      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Data |           |U|A|P|R|S|F|                               |
   | Offset| Reserved  |R|C|S|S|Y|I|            Window             |
   |       |           |G|K|H|T|N|N|                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Checksum            |         Urgent Pointer        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    Options                    |    Padding    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                             data                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Source port
  - Source port number.
- Destination port
  - Destination port number.
- Sequence number
  - 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.
- Acknowledgment number
  - If the ACK control bit is set, this field contains the value of the next sequence number which the sender of the segment is expecting to receive. Once a connection is established, this value is always sent.
- Data offset
  - 4 bits. The number of 32-bit words in the TCP header, which indicates where the data begins. The TCP header (even one including options) has a length which is an integral number of 32 bits.
- Reserved
  - 6 bits. Reserved for future use. Must be zero.
- Control bits
  - 6 bits. The control bits may be (from right to left):

U (URG)	Urgent pointer field significant.
A (ACK)	Acknowledgment field significant.
P (PSH)	Push function.
R (RST)	Reset the connection.
S (SYN)	Synchronize sequence numbers.
F (FIN)	No more data from sender.

- Window
  - 16 bits. The number of data octets which the sender of this segment is willing to accept, beginning with the octet indicated in the acknowledgment field.
- Checksum
  - 16 bits. The checksum field is the 16 bit one’s complement of the one’s complement sum of all 16-bit words in the header and text. If a segment contains an odd number of header and text octets to be checksummed, the last octet is padded on the right with zeros to form a 16-bit word for checksum purposes. The pad is not transmitted as part of the segment. While computing the checksum, the checksum field itself is replaced with zeros.
- Urgent Pointer
  - 16 bits. This field communicates the current value of the urgent pointer as a positive offset from the sequence number in this segment. The urgent pointer points to the sequence number of the octet following the urgent data. This field can only be interpreted in segments for which the URG control bit has been set.
- Options
  - Options may be transmitted at the end of the TCP header and always have a length which is a multiple of 8 bits. All options are included in the checksum. An option may begin on any octet boundary.
  - There are two possible formats for an option:
  - A single octet of option type.
  - An octet of option type, an octet of option length, and the actual option data octets.
  - The option length includes the option type and option length, as well as the option data octets.
  - The list of options may be shorter than that designated by the data offset field because the contents of the header beyond the End-of-Option option must be header padding i.e., zero.
- A TCP must implement all options.
- Data
  - TCP data or higher layer protocol.

TCP Transiction diagram



- A connection progresses through a series of states during its lifetime.
    - The states are: LISTEN, SYN-SENT, SYNRECEIVED,ESTABLISHED, FIN-WAIT-1, FIN-WAIT-2, CLOSE-WAIT, CLOSING, LAST-ACK, TIME WAIT, and the fictional state CLOSED. CLOSED is fictional because it represents the state when there is no TCB, and therefore, no connection. Briefly the meanings of the states are:

- LISTEN represents waiting for a connection request from any remote TCP and port.
- SYN-SENT represents waiting for a matching connection request after having sent a connection request.
- SYN-RECEIVED represents waiting for a confirming connection request acknowledgment after having both received and sent a connection request.
- ESTABLISHED represents an open connection, data received can be delivered to the user. The normal state for the data transfer phase of the connection.
- FIN-WAIT-1 represents waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent.
- FIN-WAIT-2 represents waiting for a connection termination request from the remote TCP.
- CLOSE-WAIT represents waiting for a connection termination request from the local user.
- CLOSING represents waiting for a connection termination request acknowledgment from the remote TCP.
- LAST-ACK represents waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request).
- TIME-WAIT represents waiting for enough time to pass to be sure the remote TCP received the acknowledgment of its connection termination request.
- CLOSED represents no connection state at all.

- A TCP connection progresses from one state to another in response to events. The events are:
- - the user calls
- - OPEN, SEND, RECEIVE, CLOSE, ABORT, and STATUS;
- - the incoming segments
- - particularly those containing the SYN, ACK, RST and FIN flags
- - the timeouts.

- The two transitions leading to the ESTABLISHED state correspond to the opening of a connection, and the two transitions leading from the ESTABLISHED state are for the termination of a connection. The ESTABLISHED state is where data transfer can occur between the two ends in both the directions.
- If a connection is in the LISTEN state and a SYN segment arrives, the connection makes a transition to the SYN_RCVD state and takes the action of replying with an ACK+SYN segment. The client does an active open which causes its end of the connection to send a SYN segment to the server and to move to the SYN_SENT state. The arrival of the SYN+ACK segment causes the client to mo ve to the ESTABLISHED state and to send an ack back to the server. When this ACK arrives the server finally moves to the ESTABLISHED state. In other words, we have just traced the THREE-WAY HANDSHAKE.
- In the process of terminating a connection, the important thing to kee p in mind is that the application process on both sides of the connection must independently close its half of the connection. Thus, on any one side there are three combinations of transition that get a connection from the ESTABLISHED state to the CLOSED state:
    - This side closes first:
- ESTABLISHED -> FIN_WAIT_1-> FIN_WAIT_2 -> TIME_WAIT -> CLOSED.
    - The other side closes first:
- ESTABLISHED -> CLOSE_WAIT -> LAST_ACK -> CLOSED.
    - Both sides close at the same time:
- ESTABLISHED -> FIN_WAIT_1-> CLOSING ->TIME_WAIT -> CLOSED.
- The main thing to recognize about connection teardown is that a connection in the TIME_WAIT state cannot move to the CLOSED state until it has waited for two times the maximum amount of time an IP datagram might live in the Inter net. The reason for this is that while the local side of the connection has sent an ACK in response to the other side's FIN segment, it does not know that the ACK was successfully delivered. As a consequence this other side might re transmit its FIN segment, and this second FIN segment might be delayed in the network. If the connection were allowed to move directly to the CLOSED state, then another pair of application processes might come along and open the same connection, and the delayed FIN segment from the earlier incarnation of the connection would immediately initiate the termination of the later incarnation of that connection.