# Network Architecture Protocols

# Contents

# What is the Internet?

Let us consider a point-to-point communication (for example, computer A to computer B). This communication only requires a means to convey the message: a line, a radio transmission, etc. If a third computer C joins the communication, then we need to increase point to point connections between A and B, A and C, and B and C.

What happens if more and more nodes join the communication network? Imagine the Internet with millions of users. In this case, point-to-point connections become unfeasible. To avoid this problem, intermediate devices come into place, like routers. In this way, Internet users only require a single communication link towards the router, which will be responsible for delivering the message to its final destination.

Thus, the Internet could be seen as a set of "nuts and bolts".

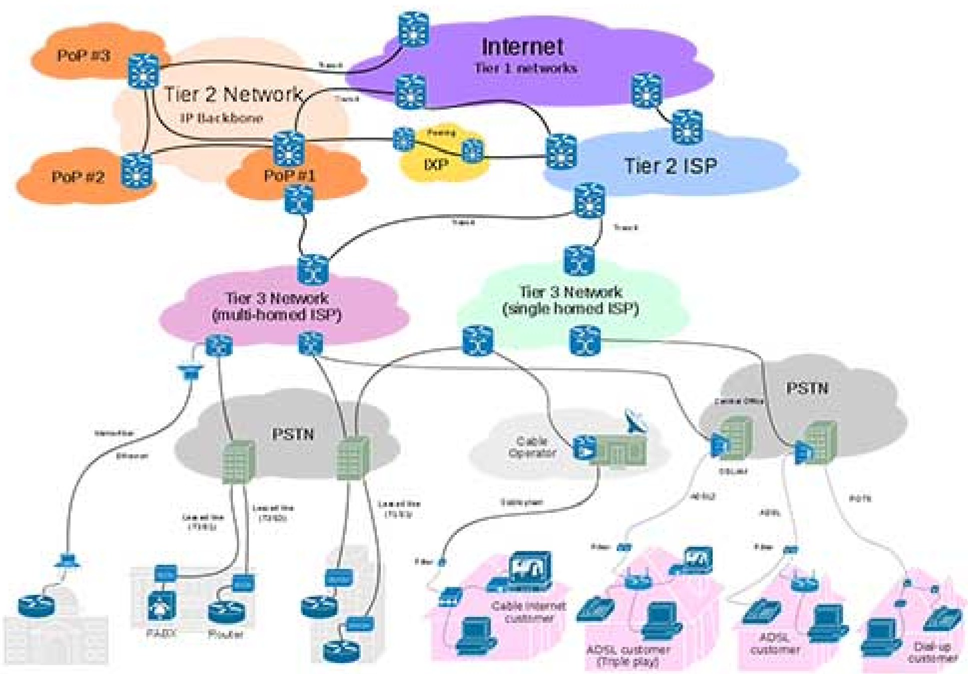



Figure 1 Internet's "nuts and bolts" view
Diagram showing how customers connect to ISP's... by Ludovic ferre... from Wikipedia CC BY-SA 3.0

In a simplified view of the Internet, the main components are:

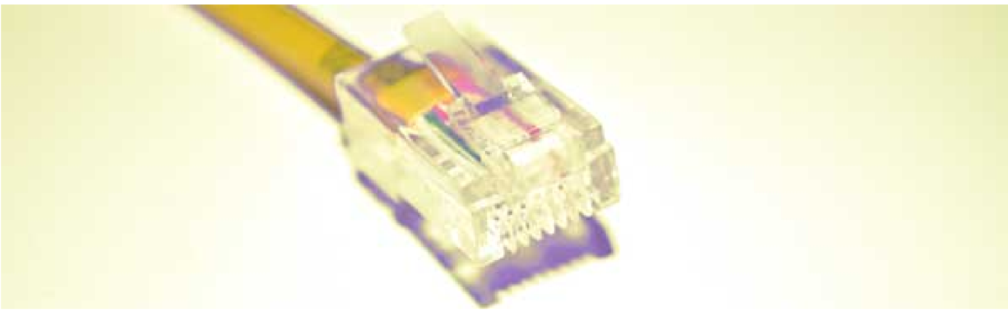
## Computing devices:

a. Hosts = End systems
   For example: Your PC, a storage server in the cloud or your neighbour's smartphone.
b. Running network applications
   For example: Skype, Instagram or, simply, your email inbox.

Picture of a computer keyboard

## Communication links:

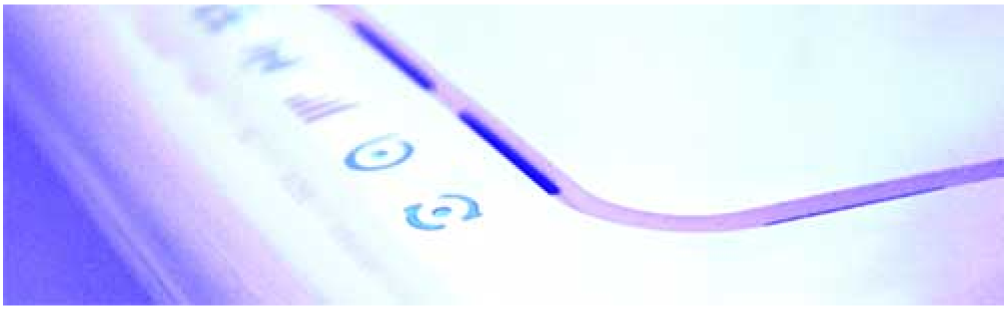
a. Fibre, copper, radio, satellite, etc.
b. Mainly characterised by their transmission rate or bandwidth



Picture of an Ethernet cable

## Forwarders or network devices:

a. Routers and switches
b. In charge of forwarding packets (chunks of data)



Picture of a home router

## Horizontal view

This horizontal view of the Internet can be divided into two main parts

### Network edge

The part of the network closest to the final users. Ranging from residential access to mobile access networks. Heterogeneous devices. An Internet Service Provider (ISP) usually provides these networks.

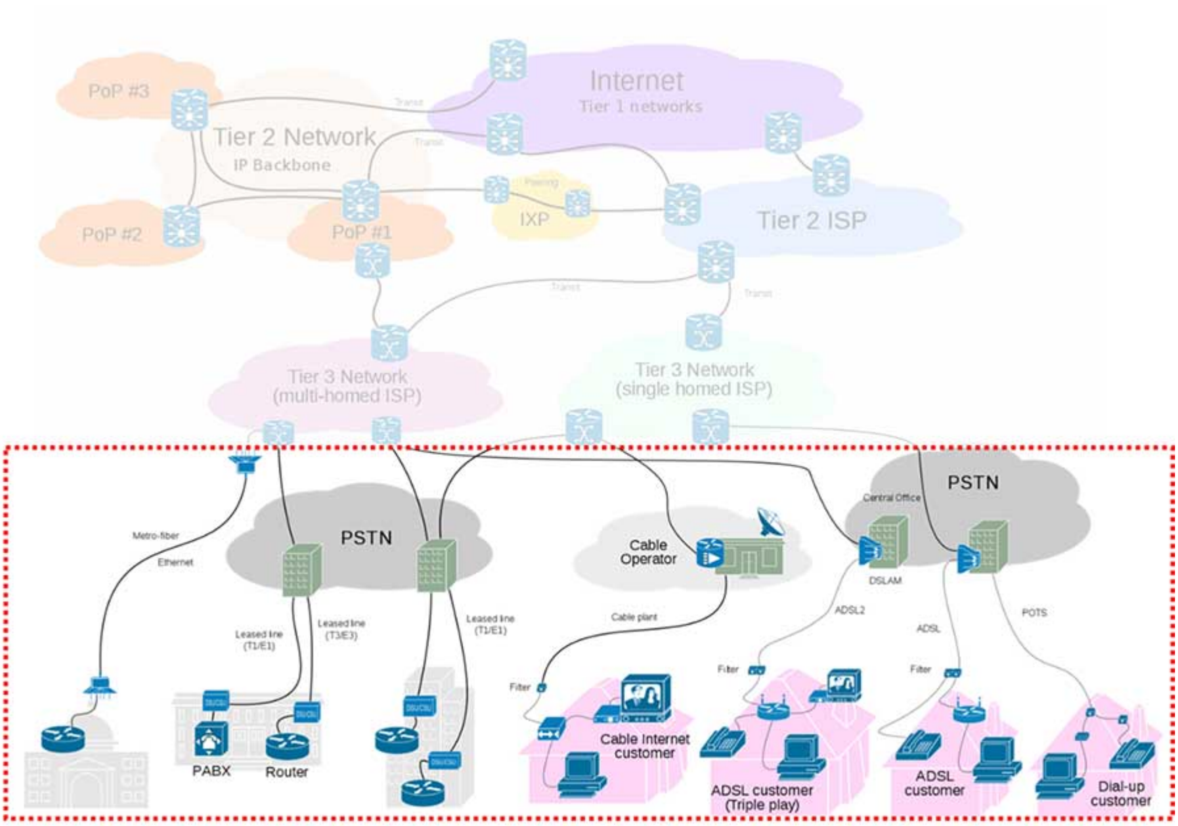



Figure 2 Network edge

## Network core

Composed of the innards of the Internet, usually “hidden” from the end user. This consists mainly of a mesh of interconnected routers. Homogeneous devices. ISPs need to coordinate among them to forward packets crossing different domains.

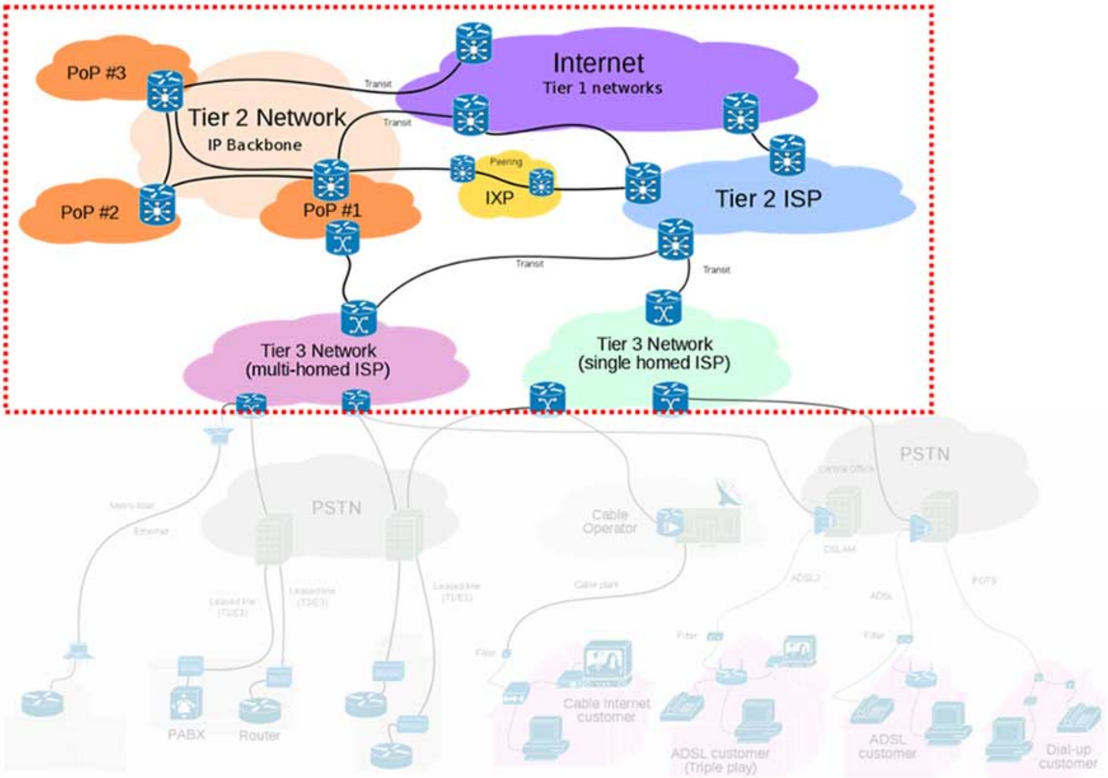



Figure 3 Network core
Diagram showing how customers connect to ISP's... by Ludovic ferre...
modified from Wikipedia CC BY-SA 3.0

# Protocol layers and service models

The vertical view of the Internet considers a different problem of communication:

- How is an email is eventually translated into bits and sent to the network media?
- How can we guarantee that the email is received, even if the communication channel is noisy and prone to errors?
- Can a huge piece of information (for example, a streaming video) be sent in just a single packet or should it be fragmented to prevent the link from collapsing?

Basically, a communication protocol needs to be defined, analogously to the human language.

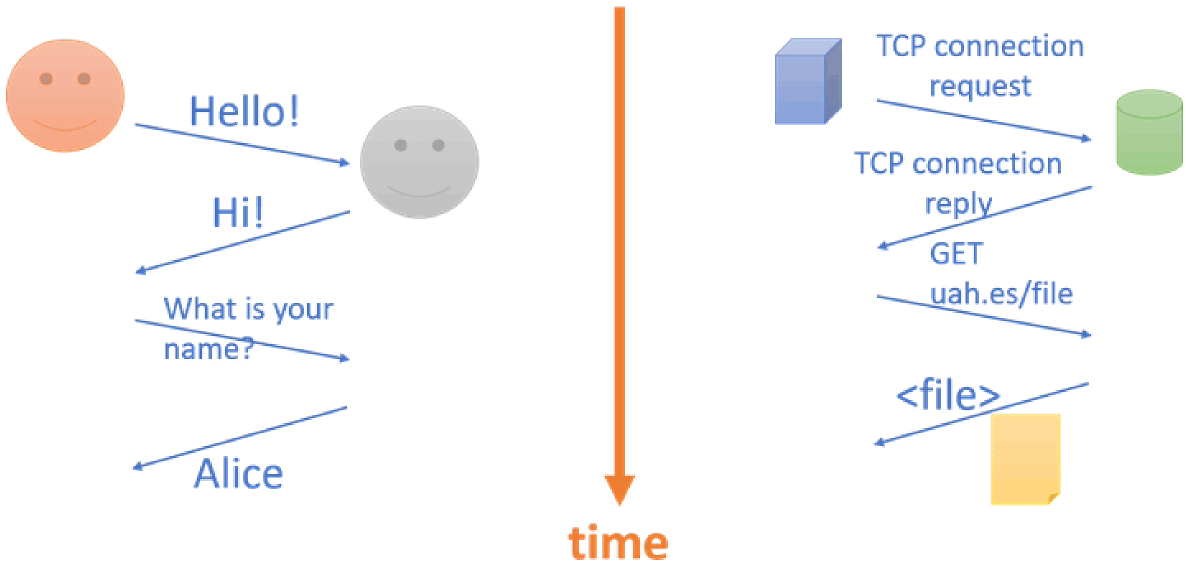



Figure 4 Human language in comparison to a network communication protocol

However, just a single protocol might be too complex to translate an email into bits. For this reason, the design is divided into protocol layers:

- Communication functions are divided into a number of modules or layers
- Network design is simplified:
    - "Big problem" is fragmented in simpler problems
        - → "divide and conquer"
    - Layered design makes things easier
    - The OSI model defines 7 layers
    - The Internet (practical) model reduces it to 5 layers
        - → The TCP/IP stack

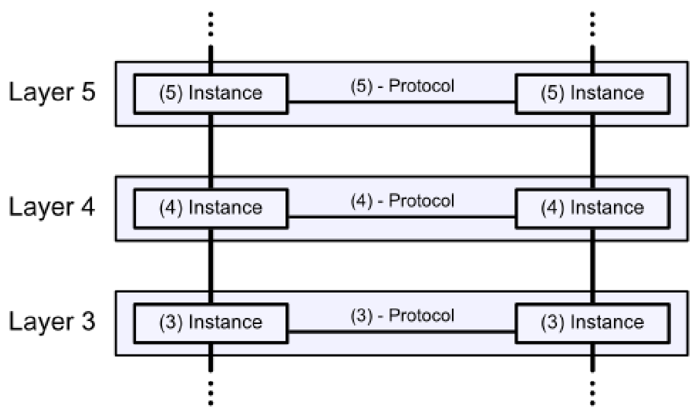



Figure 5 Protocol layers and services model diagram
Layered Communication in OSI-Model by Runtux (Own work) from Wikipedia Public Domain

# The TCP/IP model

The TCP/IP model consists of two protocols that make up the internet architecture. The TCP/IP or Internet Protocol Stack and the ISO/OSI reference model. We will look closely in more detail at these models in this section.

# The TCP/IP or Internet Protocol Stack

The TCP/IP or Internet protocol stack consists of 5 layers:

- *application*: supporting network applications
  - Web, Email, Facebook, etc.
- *transport*: process to process data transfer
  - TCP, UDP
- *network*: routing of datagrams from source to destination
  - IP
- *link*: data transfer between neighbouring network elements
  - Ethernet, WiFi, etc.
- physical: bits "on the wire"
  - In some literature, the link and physical layers are presented together, so that the TCP/IP stack remains with just 4 layers.

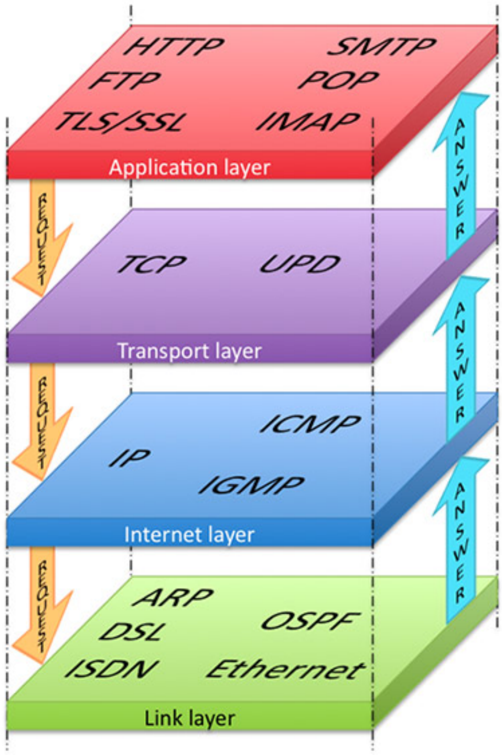



Figure 6 Internet protocol stack

A graphic representation of the Internet Protocol Stack... by Bughunter from Wikipedia Public Domain

## The ISO/OSI reference model

The ISO/OSI model defines 7 layers, the 5 mentioned above plus 2 more between application and transport:

- *presentation*: allowing applications to interpret meaning of data, such as: encryption, compression, machine-specific conventions, etc.
- *session*: synchronization, checkpointing, recovery of data exchange
- The Internet stack, as a practical approach, does not implement these layers!
    - These services, *if needed*, must be implemented in the application layer
    - The question is: are they really needed?

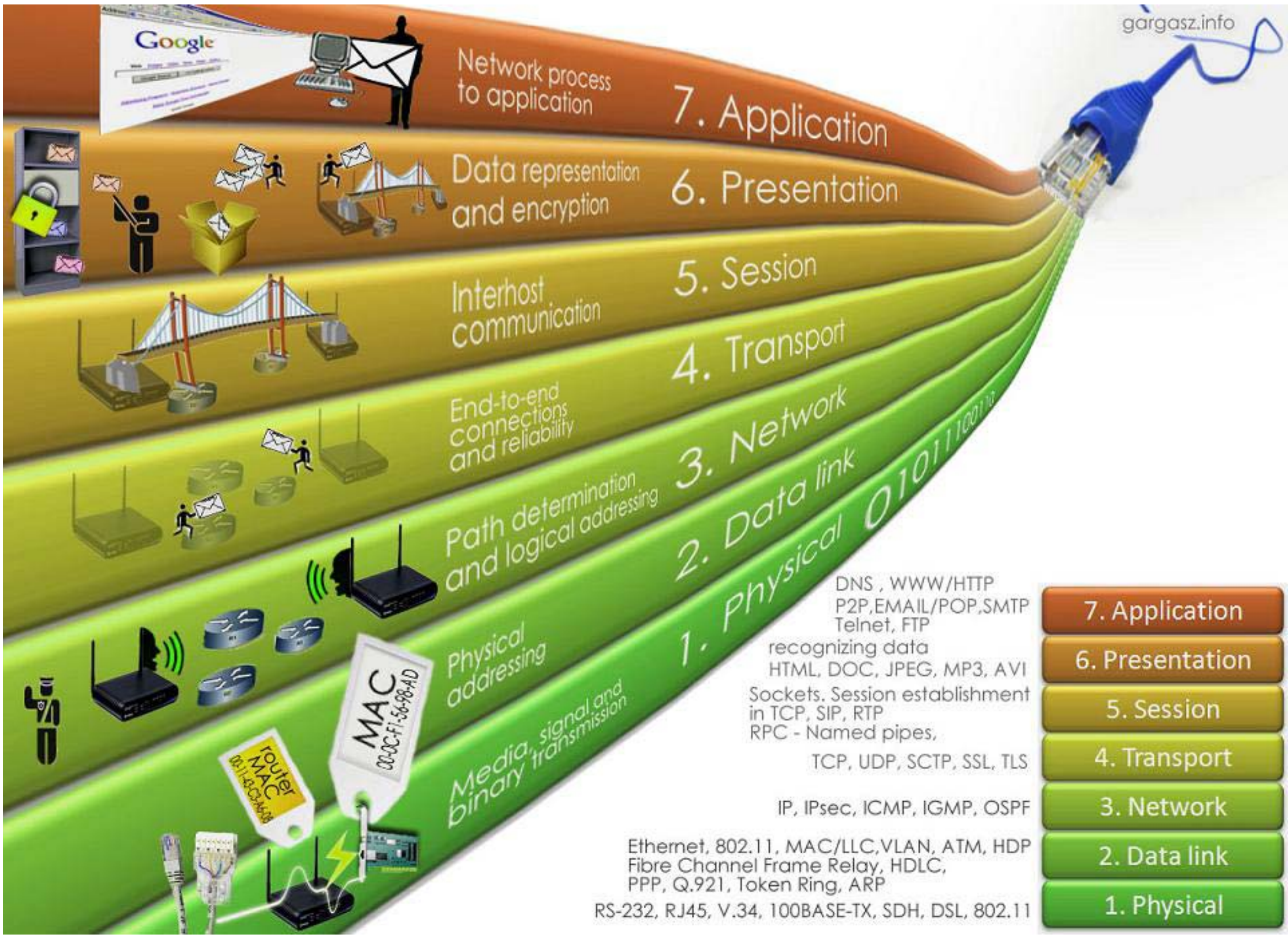



Figure 8 ISO/OSI reference model summarising services and protocols
Zbatimi i Niveleve OSI by Rexhep-bunjaku from Wikipedia CC BY-SA 3.0

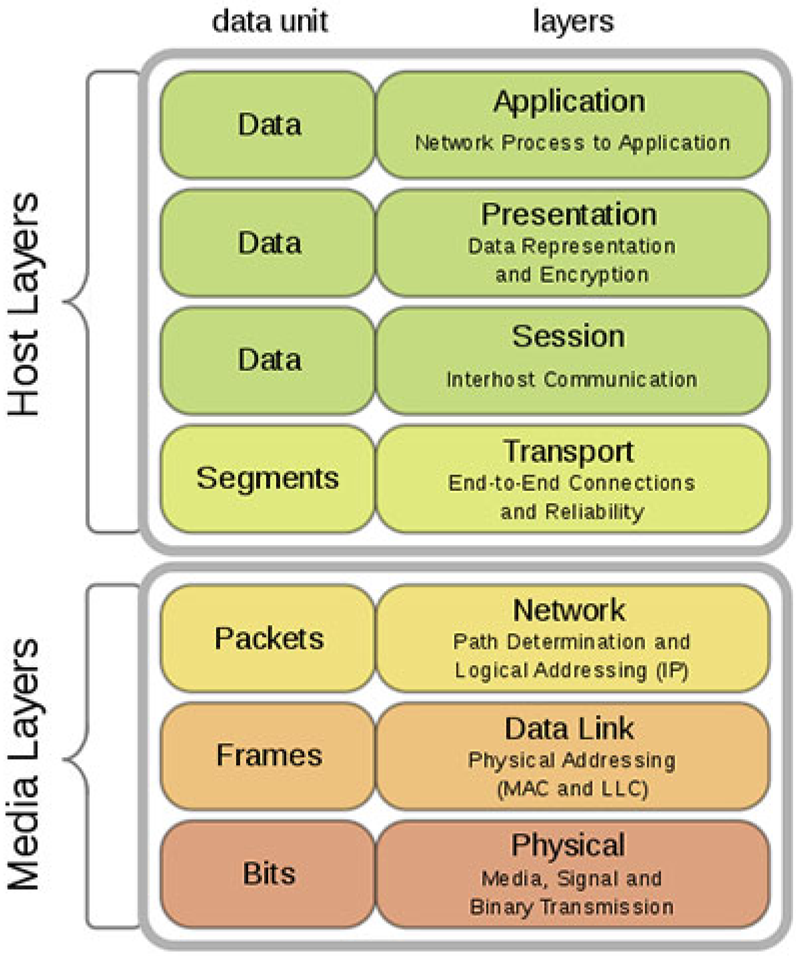



Figure 7 ISO/OSI reference model diagram
Diagram of OSI model.... by Offnfopt from Wikipedia Public Domain

Let us now examine how a message "M" would be sent from a source to a destination computer. The original message "M" is sent from the application to the transport layer. The transport layer includes its own transport header "Ht" to distinguish different messages arriving from the application layer. For example, this header could contain a sequence number or some identification number. Afterwards, it sends both the header and message to the network layer.

The network layer receives "Ht+M", although it will not be aware of what part is the original message. It then adds its own network header "Hn" and sends it down to the link layer, which finally receives "Hn+Ht+M" and adds its link layer header "Hl". The link layer sends it to the physical layer, which will not add any additional information. The physical layer is in charge of transmitting every bit of the chunk of bytes represented by "Hl+Hn+Ht+M".

This operation of adding headers to a message in each protocol layer is called packet **encapsulation**.

The receiver of these bits, will apply the opposite operation, **decapsulation**, taking out the headers for each protocol until it obtains the original message “M”.

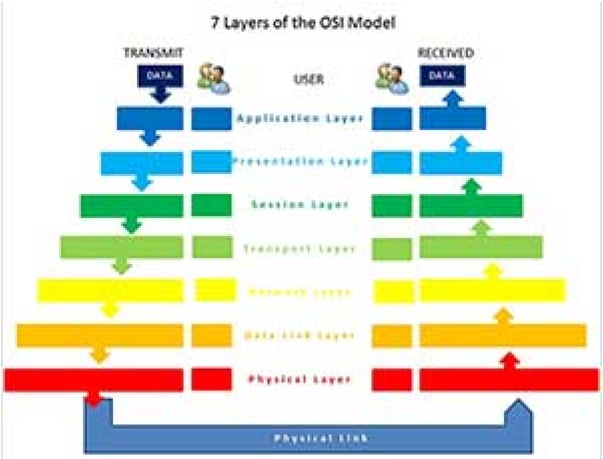



Figure 9 Encapsulation and decapsulation for a message “DATA” in the OSI model
This is an illustration of the Open Systems Interconnection Model by MrsValdry (Own Work) from Wikipedia CC BY-SA 3.0

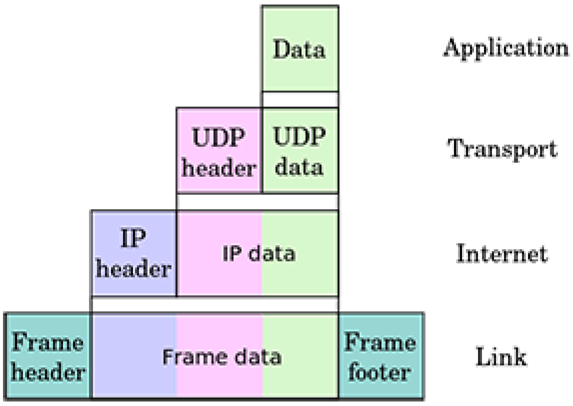



Figure 9 Encapsulation example for a message “DATA” in the TCP/IP stack, using the UDP and IP protocols Encapsulation of user data in the Unix-style UDP stack... by Cburnett from Wikipedia CC BY-SA 3.0

Notice that not all devices in the network need to implement all the protocol layers. For example, a router just implements 3 layers, as it does not need transport or application, and a switch does not even need the network layer, as it does not route packets among different networks.

## Question

If an email “M” from source contains 123 bytes, the transport header “Ht” size is 10 bytes, the network header “Hn” size is 20 bytes and the link header “Hl” size is 16 bytes.

1. What is the actual number of bits sent in the communication line towards destination? (Remember 1 byte = 8 bits)
    a. 123
    b. 169
    c. 1352
    d. 984

2. How many bytes does the destination receive at application layer level?
    a. 123
    b. 169
    c. 1352
    d. 984

## Answers

1. The answer is c. 1352
2. The answer is a. 123

# The *Application* layer

- Users interact with Application layer
    - "Users" of application layer: Software (SW) entities that use it
- Composed of:
    - SW entities (programs, processes) and services that interact with the "user" through an interface
- Among others, it includes the following protocols, and associated applications :
    - TELNET (Remote Terminal Protocol), virtual network terminal
    - FTP (File Transfer Protocol)
    - SMTP (Simple Mail Transfer protocol) → email
    - DNS (Domain Name System)
    - HTTP (Hypertext Transfer Protocol) → web

# WWW: World Wide Web → The Web

- Provides access to information, to **linked resources distributed at many hosts**
- Allows:
    - Receive (and publish) information
    - "Navigate through an ocean" of information via hyperlinks, and the interaction of different Web pages
- Operates:
    - **On demand**
        - Users receive and publish "what they want" "when they want"
    - **Client-Service Model**

- Elements of the web architecture
    - **Pages**
        - HTML, hypertext, hyperlinks
    - **Objects**
        - Can reside at the same or another host
    - **Addressing**
        - URL: Uniform Resource Locator
    - **Functional Entities**
        - Web Client (Navigator)
        - Web server
    - **HTTP Protocol**

- Addressing

- With a URL: **U**niversal **R**esource **L**ocator
  - Element for addressing resources
- URL parameters
  1. Service identifier (protocol)
  2. Domain name of Server Host
  3. Path to access object
  4. Object name
- URL syntax
  - Characters "://" separate first parameter from second one
  - Character "/" used to separate the rest of parameters


Figure 10 Example of URL showing the four types of parameters that usually compose it

- Web Client (**Browser**)
  - Includes:
    - User agent
      - Software that provides an interface between the user and Web application
      - Captures and visualizes pages and allows navigation through their contents
      - Diverse configuration characteristics
    - HTTP Client
      - Communication software that implements the HTTP Client part
    - Best known browsers:
      - Chrome
      - MS Internet Explorer / Edge
      - Firefox

# HTTP: Hyper Text Transfer Protocol

- **Protocol for transfer of resources**
  - Resources:
    - Files
    - Result of :
      - Database query
      - Program execution
      - Automatic translation of a document
- Specified in two incompatible versions:
  - HTTPv1.0 → RFC 1945
  - HTTPv1.1 → RFC 2616
- Client-Server Model
- Operates with **simple "transactions"**
  - Request – Response
    - Client requests objects → generates "requests"
      - Messages in ASCII format (understandable for humans
    - Server transfers objects → generates "responses"
      - ASCII messages MIME type (RFC 822)
  - **Stateless** and **Connectionless protocol**
    - Server does not store information on clients or transactions
    - Simplifies Servers design ⎝ High performance web servers, capable of handling many TCP connections
- Uses TCP as transport layer protocol:
  - Client **initiates a TCP connection** (creates socket) to server, **port 80**
  - **Server accepts TCP connection** from client (connection socket is created in server)
  - **HTTP messages** (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
  - TCP connection closed

- HTTP response status codes
  - **200 OK**
    - request succeeded, requested object later in this msg
  - **301 Moved Permanently**
    - requested object moved, new location specified later in this msg (Location:)
  - **400 Bad Request**
    - request msg not understood by server
  - **404 Not Found**
    - requested document not found on this server
  - **505 HTTP Version Not Supported**

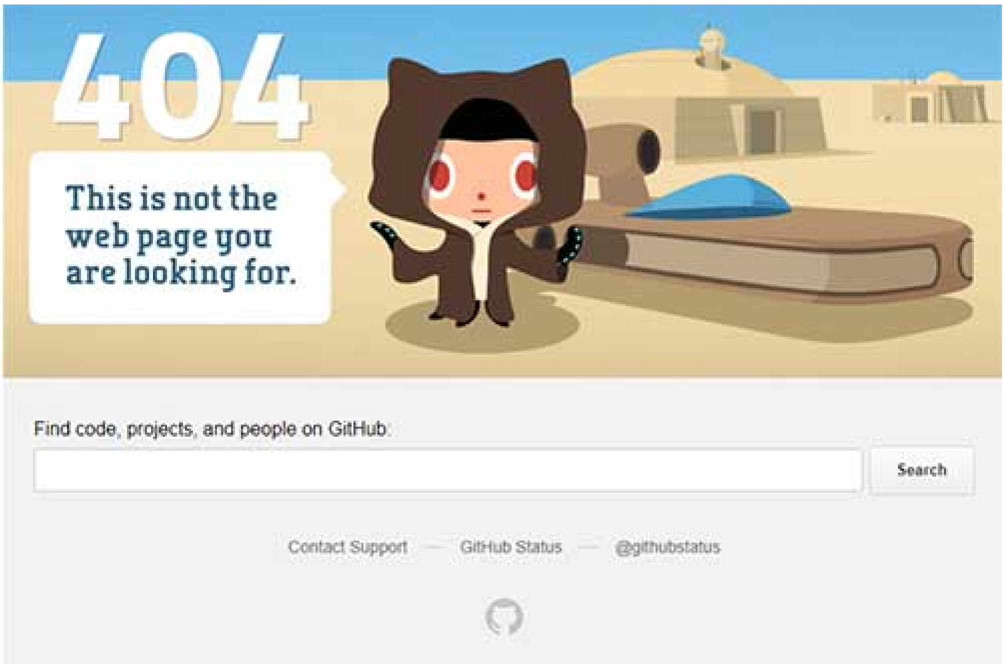



Figure 11 Example of HTTP status code 404 from GitHub.com

## Question

Regarding HTTP:

1. If you are seeing this web page right now, what status code have you received?
   a. 404
   b. 200
   c. 400
   d. 301

2. What transport layer protocol uses?
   a. VDP
   b. UDP
   c. CDP
   d. None of the above

## Answers

1. The answer is b. 200
2. The answer is d. None of the above.

# DNS: Domain Name Service

- Directory Service for Internet resources
    - Resources?
        - IP addresses
        - Host Info
        - Canonical names of Host
        - Simplified names (aliases) Host
        - Names of mail servers
        - Other

- What is a Domain Name?
    - Initial specifications in RFC 1034 and 1035
        - **Method of identifying resources on the Internet**, for persons
        - Symbolic name **easy to remember**
        - **Chain of "tags"**
    - They can identify:
        - a **host**. E.g. Host "cisneros" cisneros.aut.uah.es
        - an **organisation**. E.g. for the Department of Automation: automatica.aut.alcala.es
        - a **service**. E.g. UAH Web: www.uah.es

What are the functional entities of DNS?

- The DNS system is based on a model of the type:
    - **Client-Server**, and **Distributed** (not centralized)
- Transport Service used
    - Normally **UDP** port 53 by default
    - It could also be TCP, port 53
- Functional Entities
    - DNS **clients** (resolvers)
        - Typically located in Hosts
        - Serve certain Network Applications
    - DNS **servers**
        - As applicable, may act:
            - Just as Servers, or mixed, as a client and server → Proxy

## The *Transport* layer

- Manages communication between Transport entities “end to end”
    - → Transparency of network nodes
- Main protocols:
    - TCP : Connection Oriented (CO)
    - UDP: Connectionless (CL)
- Other protocols (specified later):
    - SCTP (**S**tream **C**ontrol **T**ransport **P**rotocol), RFC 2960. Created initially to transport telephone signalling over IP networks
    - RTP*/RTCP (Real Time Transport Protocol), RFC´s 1889, 3550. For transport of information with real time content:
        - Voice, video.

## TCP (Transmission Control Protocol)

- Defined by RFC’s 793, 1122, 1323
- Connection Oriented Protocol
- Provides **reliable end to end transport** of data ”segments”
- Functions:
    - Fragments, if needed, the data sequence received from Application layer and forwards them to the Inter networking layer (IP) as several segments.
    - Reassembles it at destination (reorders and asks for retransmissions)
    - Flow and error control
    - Congestion control

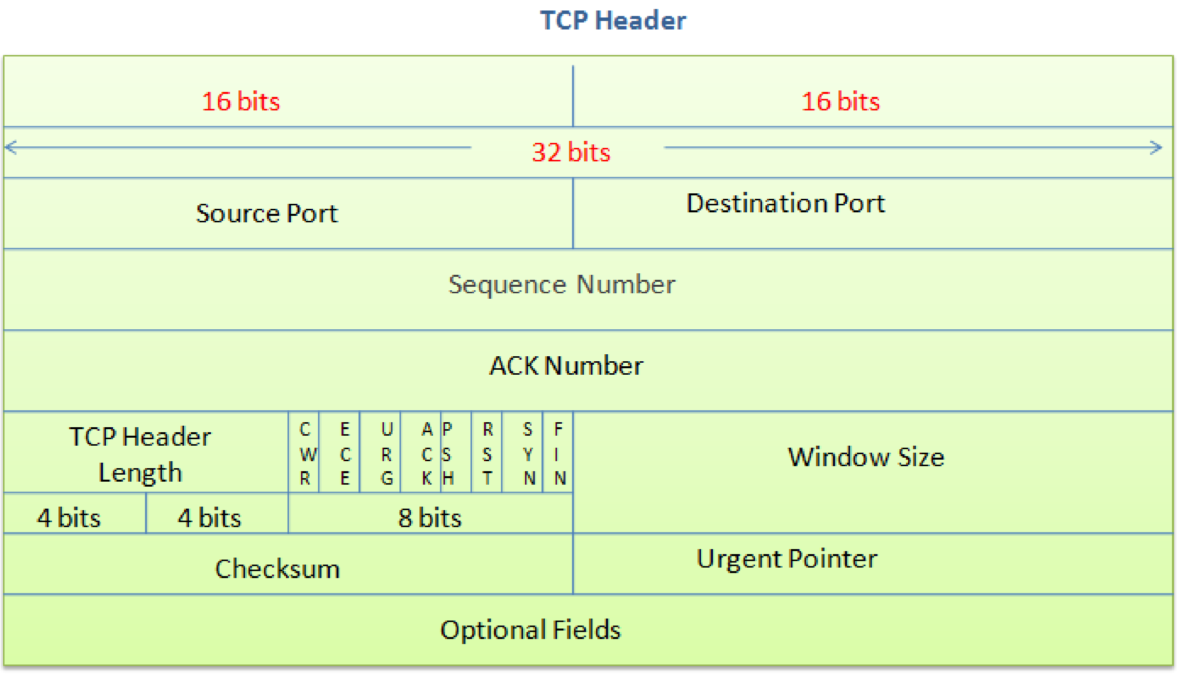



Figure 12 TCP segment format
TCP Header by Sajidur89 (Own work) from Wikipedia CC BY-SA 3.0

## Header of UDP

- Defined at RFC 768
- Connectionless
- Provides **unconfirmed end to end transport service of datagrams between applications**
- **Very simple protocol:**
  - No flow control, congestion control, error control
- Suitable for applications:
  - That do not require sequenced delivery or flow control
  - Real time (RT)
  - To distribute information to multiple destinations

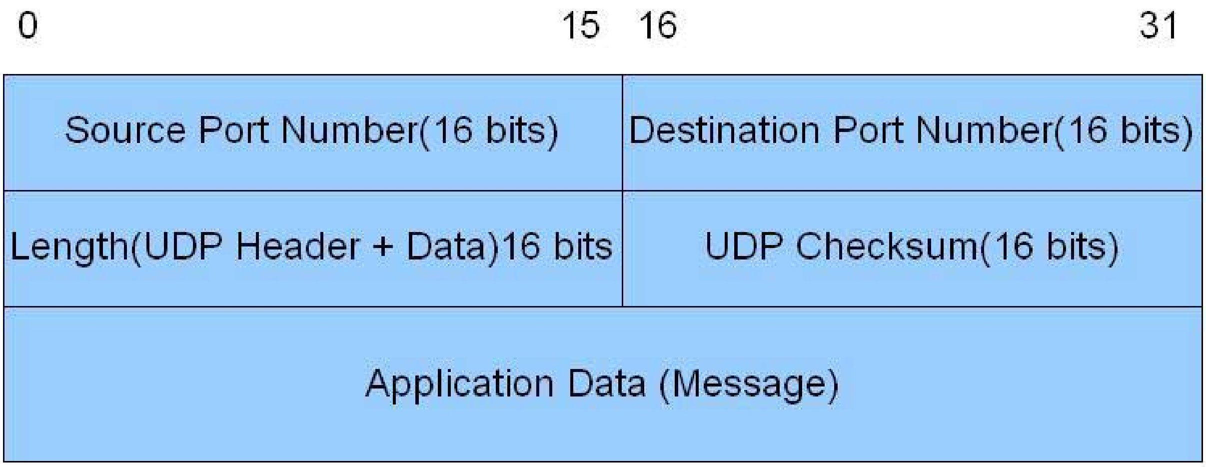



Figure 13 UDP segment format
Header of UDP by Devarshi from Wikipedia CC BY-SA 2.5

## Summary

### You should now be able to:

- Describe the components of the Internet architecture from both a horizontal and a vertical point of view
- Tell the difference between the edge and the core network
- Explain why the network design is divided into protocol layers and what a network service is
- Enumerate the layers of the TCP/IP stack or the ISO/OSI reference model
- Define the concept of encapsulation in communication networks
- Describe the characteristics and name some protocol examples of the Application layer
- Describe the characteristics and name some protocol examples of the Transport layer

## Further reading


- J. Kurose & K.W. Ross. *Computer networking: A top-down approach*, 7th Ed., Pearson Education, 2017. Chapters 1 to 3.