

What is a protocol?

A **protocol** is a standard method which enables communication between processes (potentially running on different machines), i.e. a collection of rules and procedures to be observed for issuing and receiving data over a network. There are several protocols according to how the communication is expected. Some protocols for example will specialise in the exchange of files (FTP), others may be used simply to manage the status of transmission and errors (as is the case with ICMP), ...

On the Internet, the protocols used belong to a suite of protocols, or a collection of linked protocols. This suite of protocols is called [TCP/IP](#).

Among others, it contains the following protocols:

- [HTTP](#)
- [FTP](#)
- [ARP](#)
- [ICMP](#)
- [IP](#)
- [TCP](#)
- [UDP](#)
- [SMTP](#)
- [Telnet](#)
- [NNTP](#)

Connection oriented and connectionless protocols

Generally protocols are classed in two categories depending on the level of data monitoring required:

- **Connection oriented protocols:** These protocols operate data transmission monitoring **during** a communication established between two machines. In such a schema, the recipient machine sends delivery acknowledgements during communication, so the originator machine is responsible for the validity of data that it sends. Data is therefore sent in the form of data flow. [TCP](#) is a connection oriented protocol
- **Connectionless protocols:** This is a communication method in which the originator machine sends data without warning the recipient machine, and the recipient machine receives the data without sending a delivery notification to the originator. Data is therefore sent in the form of blocks (datagrams). [UDP](#) is a connectionless protocol

Protocol and implementation

A protocol uniquely defines the way in which machines must communicate, i.e. the format and sequence of data to be exchanged. Conversely, a protocol does not define the way that software is programmed in such a way that it is compatible with the protocol. This is called **implementation** or the translation of a protocol into a [programming language](#).

Protocol specifications are never exhaustive; also it is usual that implementations are subject to a certain interpretation of the specifications, which sometimes leads to the specificities of certain implementations or worse to incompatibility or flaws in security!