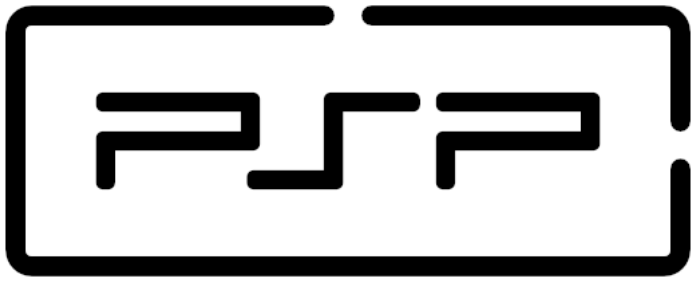


Process and Service Programming

4.1 TCP IP protocol stack

4.1 TCP IP protocol stack

4.1.1. TCP/IP Layers

The TCP/IP Stack, or the internet protocol suite, is a set of communication protocols used by the Internet or similar networks.

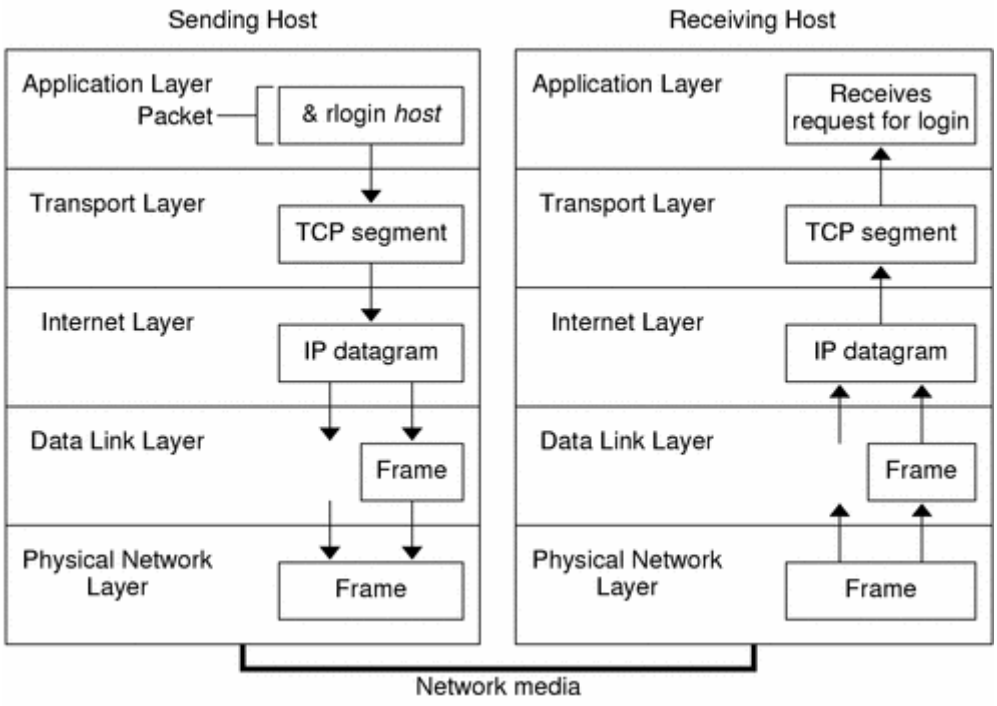
TCP/IP is the world's most widely-used non-proprietary protocol suite because it enables computers using diverse hardware and software platforms, on different types of networks, to communicate. The protocols work equally well in both LANs and WANs.

TCP/IP is a collection of protocols named after its two best-known and most important protocols, the `Transmission Control Protocol (TCP)` and the `Internet Protocol (IP)`. As well as these relatively low-level protocols, TCP/IP includes several higher level protocols that facilitate common applications such as electronic mail, terminal emulation, and file transfer.

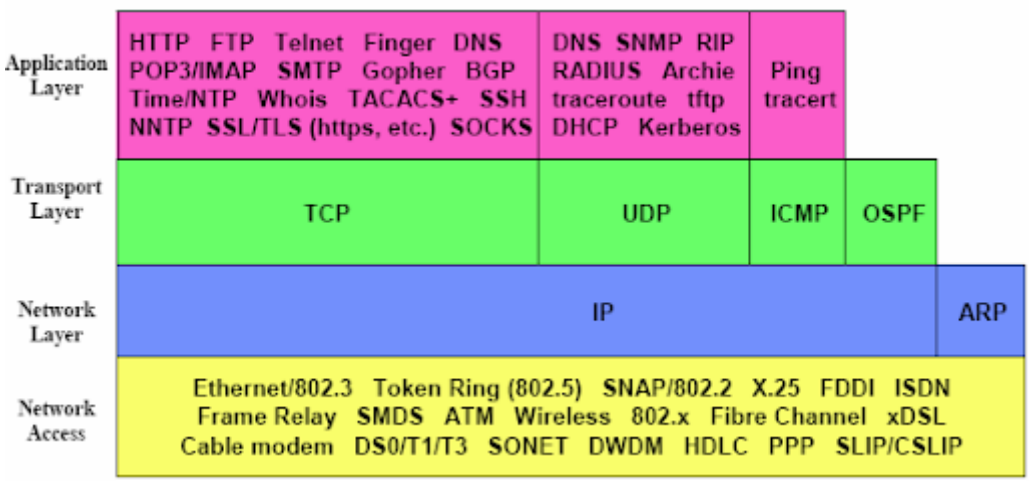
> **Protocol RFC**
>
> Each Internet protocol, together with any subsequent amendments, is described in a document known as a Request For Comments (RFC).
>
> A list of RFCs is available at: http://www.ietf.org/rfc.html.

It is called a **stack** because it is typically designed as a hierarchy of layers, each supporting the one above it and using those below it. Each layer solves a specific set of problems involving the transmission of data and provides well-defined services to the layers above it.

The TCP/IP model has four layers. From lowest to highest, these are the link layer, the internet layer, the transport layer, and the application layer, as shown below.



- The **link layer** provides the interface with the underlying network hardware and physical wired or wireless connection media.
- The **internetwork layer** provides addressing and routing functions that ensures messages are delivered to their destination. Internet Protocol (IP) is the most important protocol in this layer and probably on the whole stack.
- The **transport layer** oversees the end-to-end transfer of data, and can handle a number of data streams simultaneously. The main transport layer protocol is `Transmission Control Protocol (TCP)`, which provides a reliable, connection-oriented service. `User Datagram Protocol (UDP)` provides an unreliable, connectionless service.
- An **application layer** protocol is specific to a particular type of application (e.g. file transfer, electronic mail, network management etc.) and is sometimes embodied within the application's client software, although it could also be implemented within the operating system software. The interface between an application layer protocol and a transport layer protocol is defined with reference to `port numbers` and `sockets`.



4.1.2. Addresses and ports - Sockets

IP Addresses

Each host on a TCP/IP network is assigned a unique IP address consisting of a network number and a host number. The network number identifies a specific network. The host number identifies a host on a network and is assigned by the network administrator.

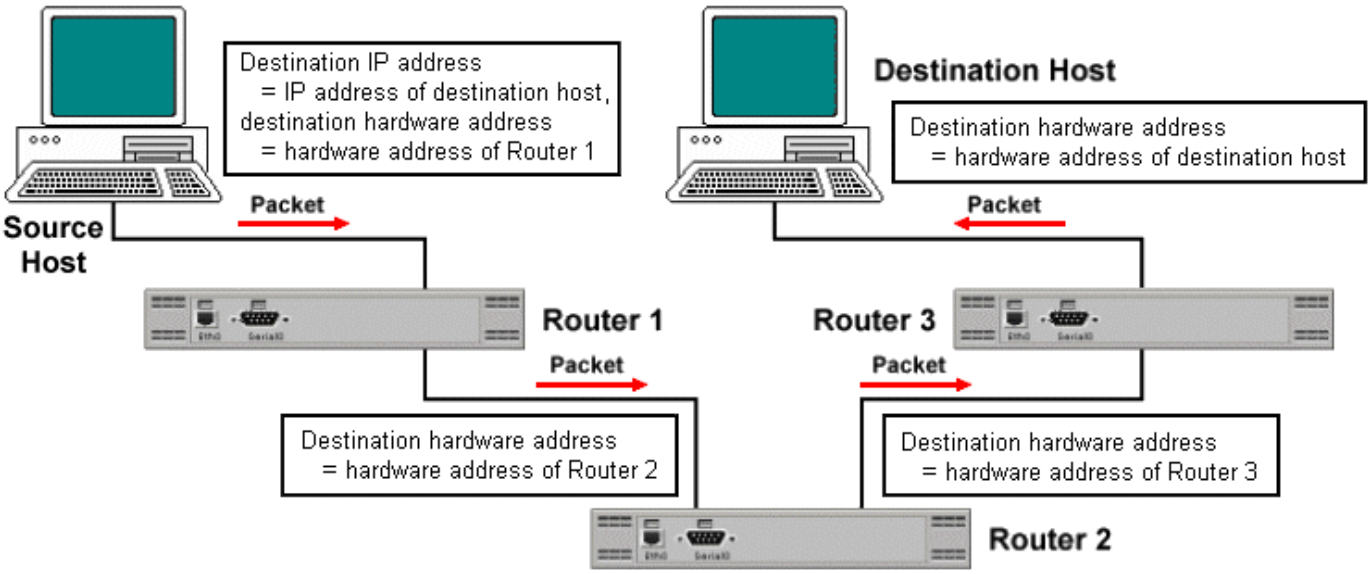
Ipv4 IP address are 32-bit addresses. The IP address is grouped into four binary octets (an octet is a group of eight bits) and is represented using dotted decimal notation. The minimum value for an octet is 0, and the maximum value is 255.

> 192.168.0.100
> 127.0.0.1
> 10.1.100.1

IPv6 addresses are typically composed of a 64-bit network prefix, and a 64-bit host part. IPv6 addresses are normally written as eight groups of four hexadecimal numbers. A group consisting solely of zeros can be omitted. Leading zeros in a group can also be omitted.

So the addresses below are all valid and equivalent to each other

> 2001:0db8:0000:0000:0000:0000:1428:57ab
> 2001:0db8:0000:0000:0000::1428:57ab
> 2001:0db8:0:0:0:0:1428:57ab
> 2001:0db8:0:0::1428:57ab
> 2001:0db8::1428:57ab
> 2001:db8::1428:57ab



Ports

An application process running on one computer that wants to communicate with an application process running on another computer identifies itself using a 16-bit port number, which is subsequently used by the transport layer protocol (TCP or UDP) to deliver incoming messages.

Port numbers go from 0 to 65535, and they are grouped in three ranges

Port group	Port range	Description
Well know ports or system ports	0 - 1023	Used by standard protocols and services
Registered ports	1024- 49151	Reserved by companies and organizations for their own services
Ephemeral ports	49152 - 65535	For free use by clients and servers

Common server applications such as Telnet and FTP use one or more of the well known port numbers (these range between 1 and 1023). Most server applications only use one port, although some (like FTP) use two. The use of a specific port number by server applications allows the client process to send a request to a server without having to first find out which port is being used by the server application.

> HTTP requests, for example, are addressed by default to port 80 on the server.

The clients themselves do not need to use a well known port, since they are initiating the communication. A client process is dynamically allocated a port number (in the range 1024 to 65535) by the client operating system. This number is subsequently included in all datagrams sent to the server.

Sockets

A `socket` is essentially an addressable end point in a communication between two processes, and consists of a **unique combination of IP address, port number and transport layer protocol (usually TCP)**.

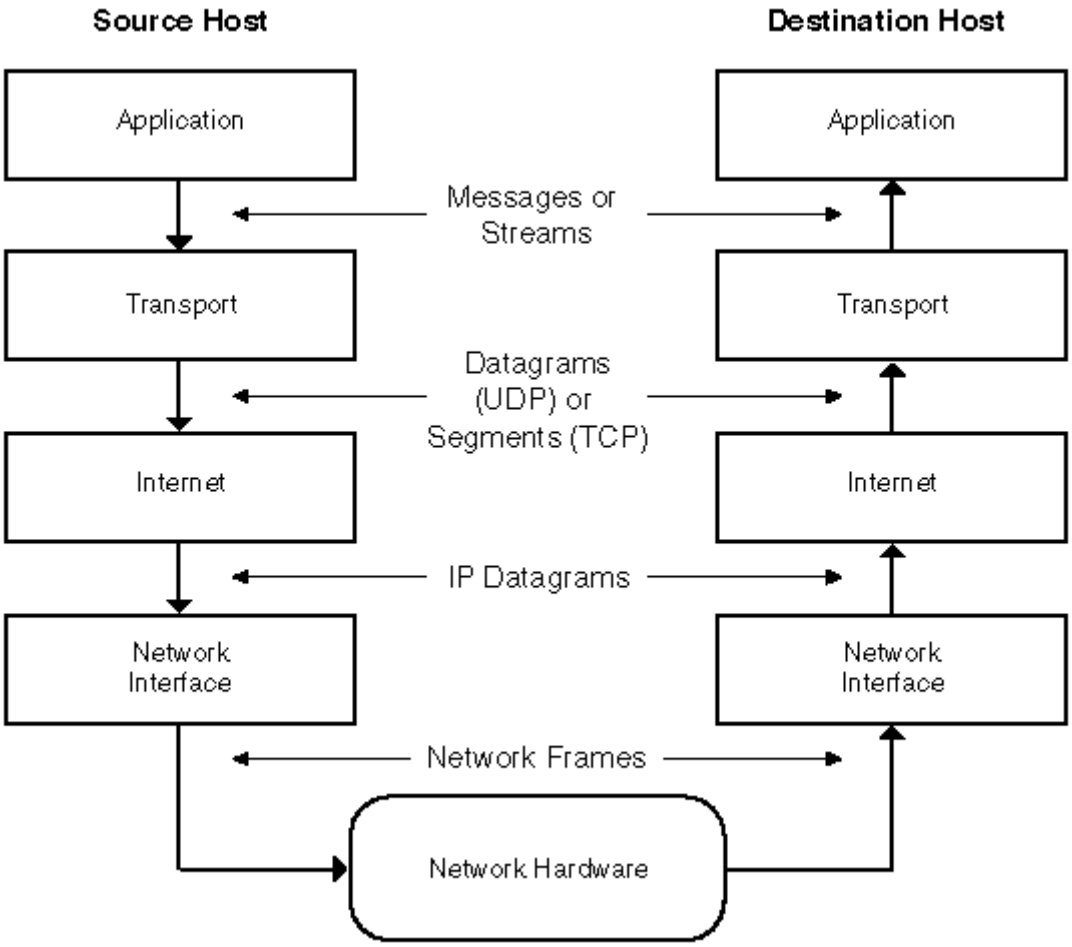
When a client application wishes to communicate with a server application, the operating system creates a socket which is then used by the client application to receive incoming data from the server. The unique combination of transport protocol, IP address and port number allows the communication end point to be addressed by a process running on a remote server, and ensures that data is delivered to the process for which it is intended.

The server application will have its own socket for communicating with the client, and a connection is established between client and server using the two socket addresses. The applications exchange information by writing to, or reading from, the sockets they have created.

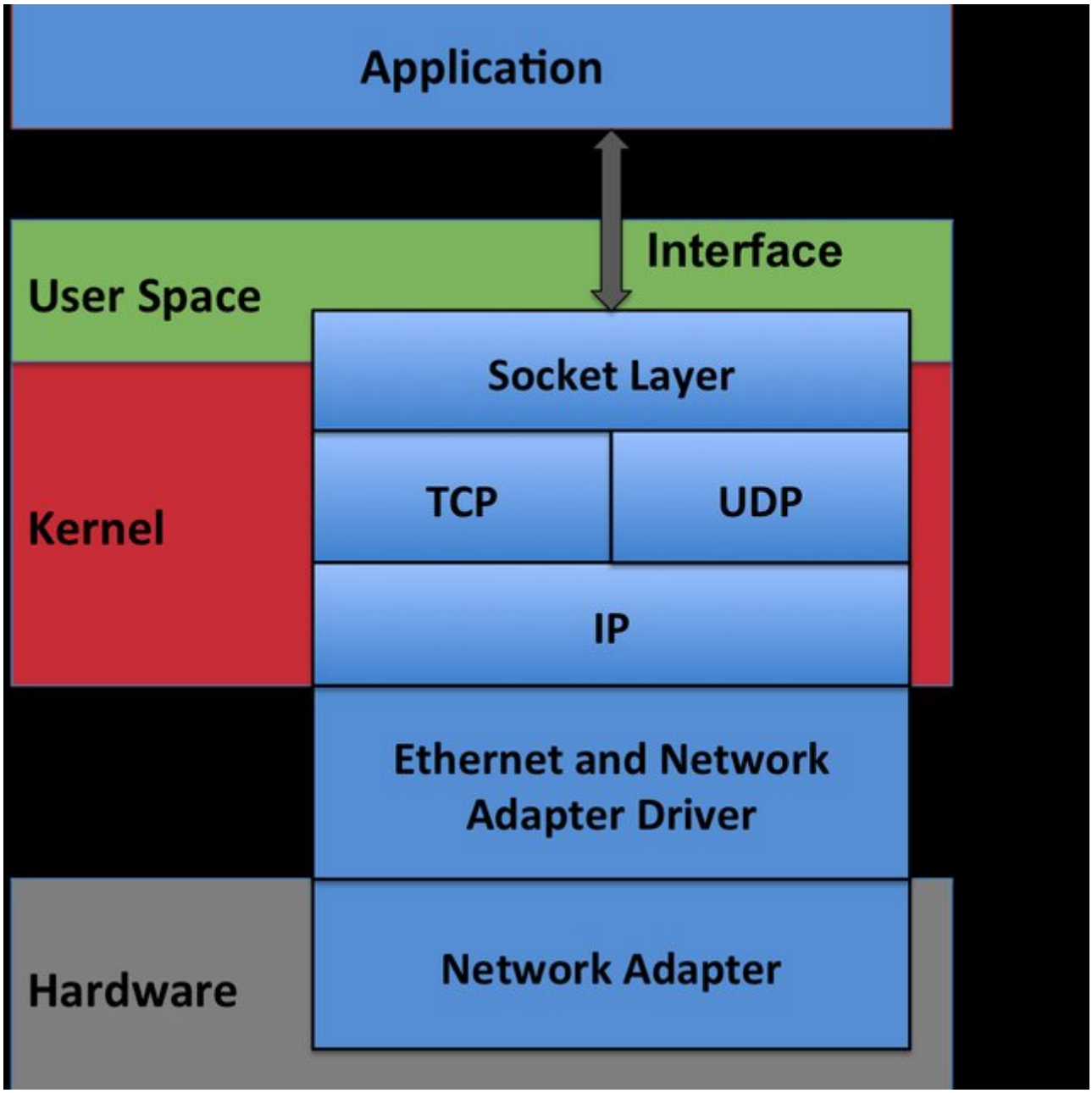
The connection used by a client process consists of two sockets, one at each end of the connection. The connection can thus be identified by a unique combination of four numbers - the source and destination IP addresses, together with the source and destination port numbers.

It is possible for several client applications running on different computers to connect to the same destination socket on a server. So there is no confusion as to which computer a datagram is destined for, even if the source and destination ports are the same in each case.

Using sockets, it is even possible for several client applications running on the same computer to connect to the same destination socket on a server. Datagrams sent to the client by the server contain the socket address for each client process, which includes the client process's individual port number, so there is no confusion as to which process a datagram is bound for.



As the real communication flow goes from one layer to the next or previous one, applications on each layer are abstracted from the underlying layers, so their communications flows to the same layer on the other side. In the TCP/IP stack different pieces of information are managed at each level as shown in the previous diagram.



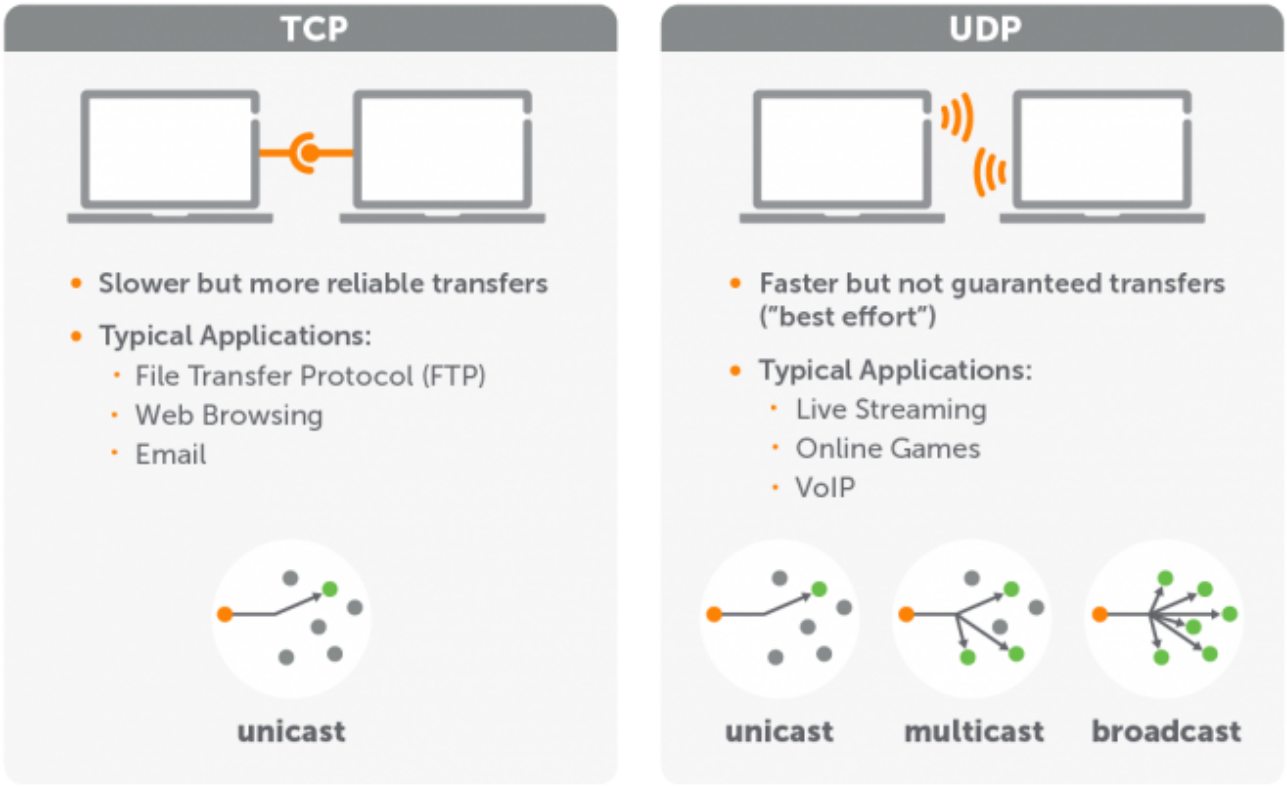
As stated before, sockets are the bridge between application layer and transport layer. That's the point where our applications are going to be developed and run, giving service to higher level protocols.

4.1.3 TCP vs UDP

The `Transmission Control Protocol (TCP)` is a widely used connection-oriented transport layer protocol that provides reliable transfer of data between two end points, and includes mechanisms to handle flow-control, segmentation, error recovery, and multiplexing.

The TCP transport service offers the following features:

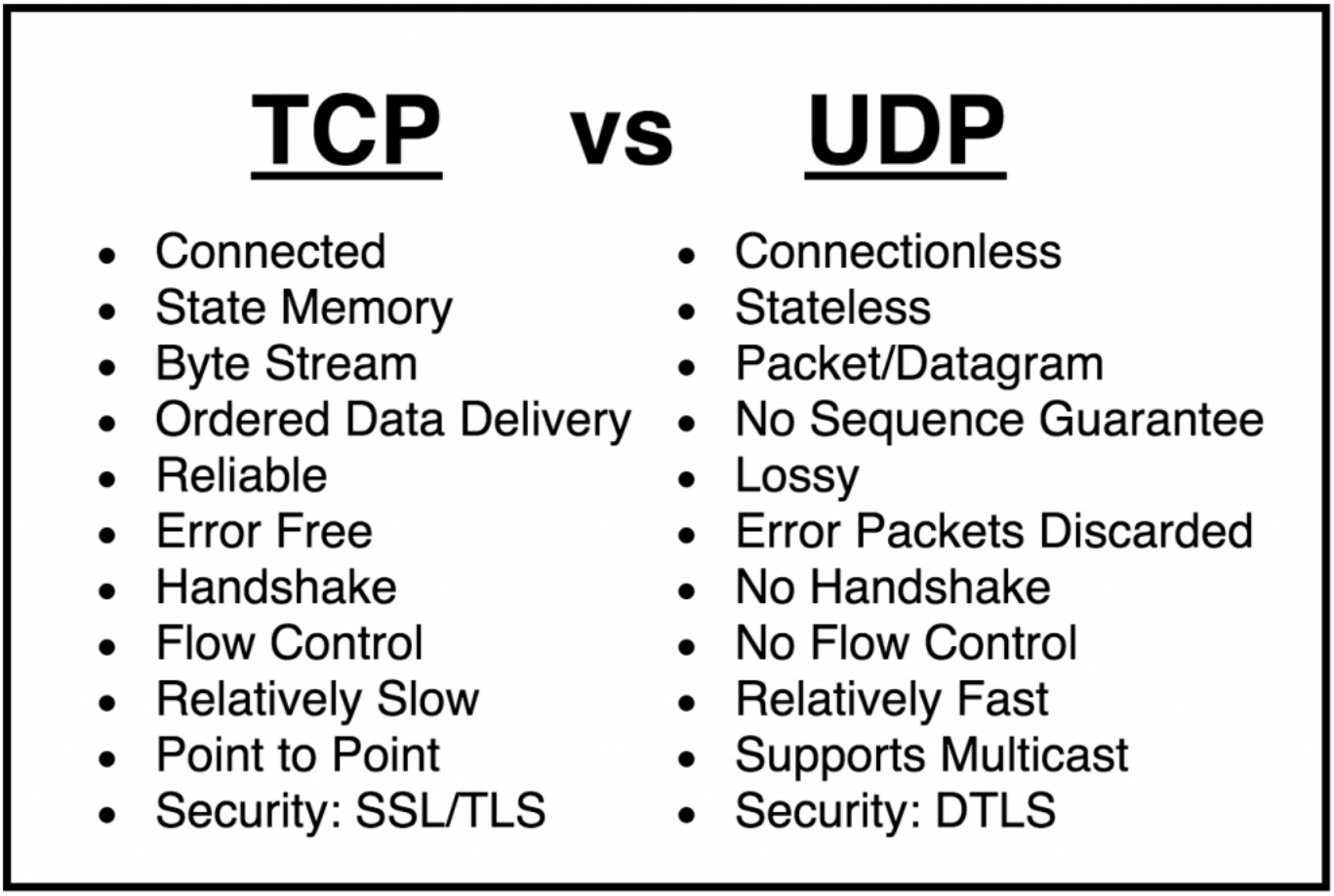
- Full duplex communication: both ends of a connection can transmit simultaneously
- Timing: timers are used to ensure that data is transmitted in a timely fashion
- Sequencing: message blocks are given sequence numbers to enable messages to be reassembled in the correct order before being passed to the application layer protocols on the destination computer
- Flow control: the flow of data is regulated using buffers and windows
- Error handling: checksums are provided to enable transmission errors to be detected and dealt with



The `User Datagram Protocol (UDP)` is an unreliable, connectionless protocol that works at the transport layer of TCP/IP, and provides a datagram delivery service to applications with a minimum of overhead. UDP provides a very simple interface between the application layer and the internetwork layer.

UDP does not provide any guarantee of delivery, nor does it provide error recovery or flow control. No connection is established, and hence no handshaking procedure is required. Packets may arrive out of order, not arrive at all, or be duplicated.

UDP is the transport protocol for a variety of application-layer protocols, including Simple Network Management Protocol (SNMP), Dynamic Host Configuration Protocol (DHCP), Routing Information Protocol (RIP), and the Domain Name System (DNS), as well as streaming media applications such as Voice over IP (VoIP).


TCP vs UDP
Connected
State Memory
Byte Stream
Ordered Data Delivery
Reliable
Error Free
Handshake
Flow Control
Relatively Slow
Point to Point
Security: SSL/TLS
Connectionless
Stateless
Packet/Datagram
No Sequence Guarantee
Lossy
Error Packets Discarded
No Handshake
No Flow Control
Relatively Fast
Supports Multicast
Security: DTLS