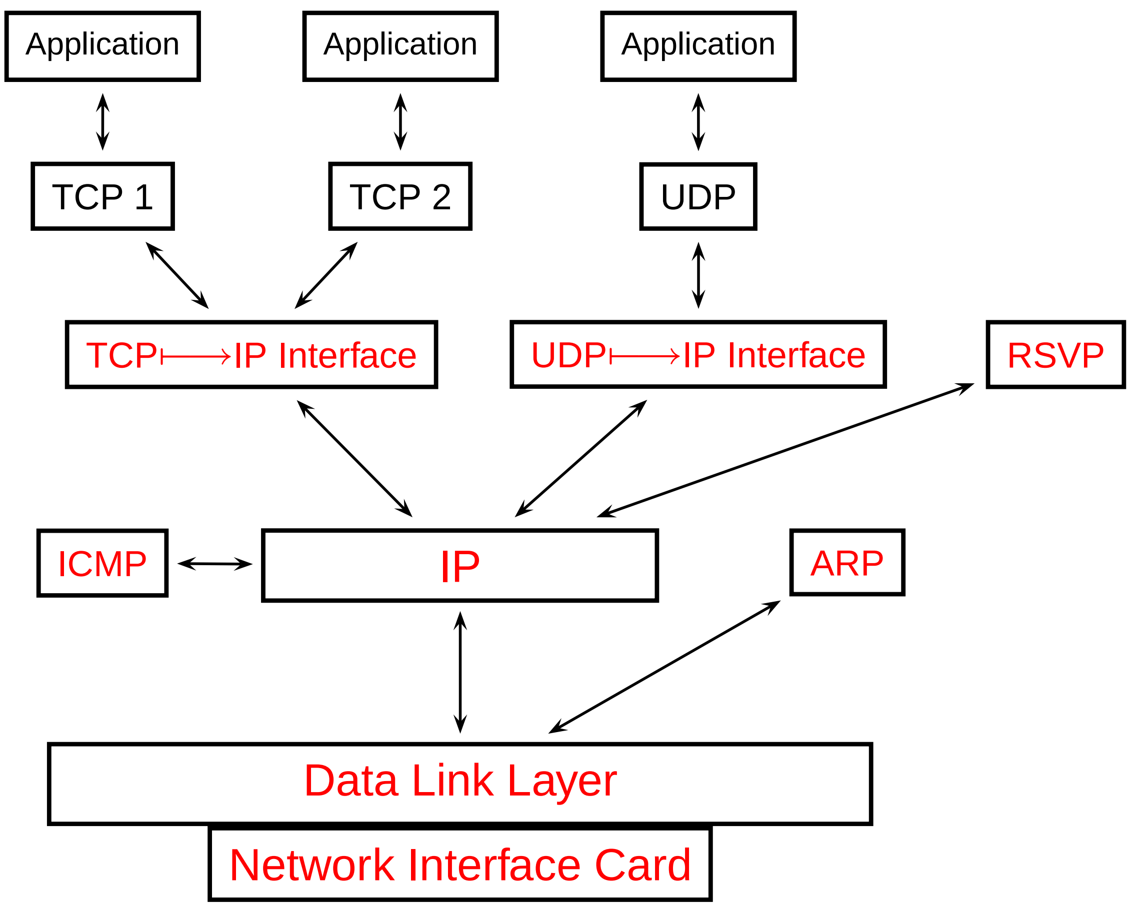
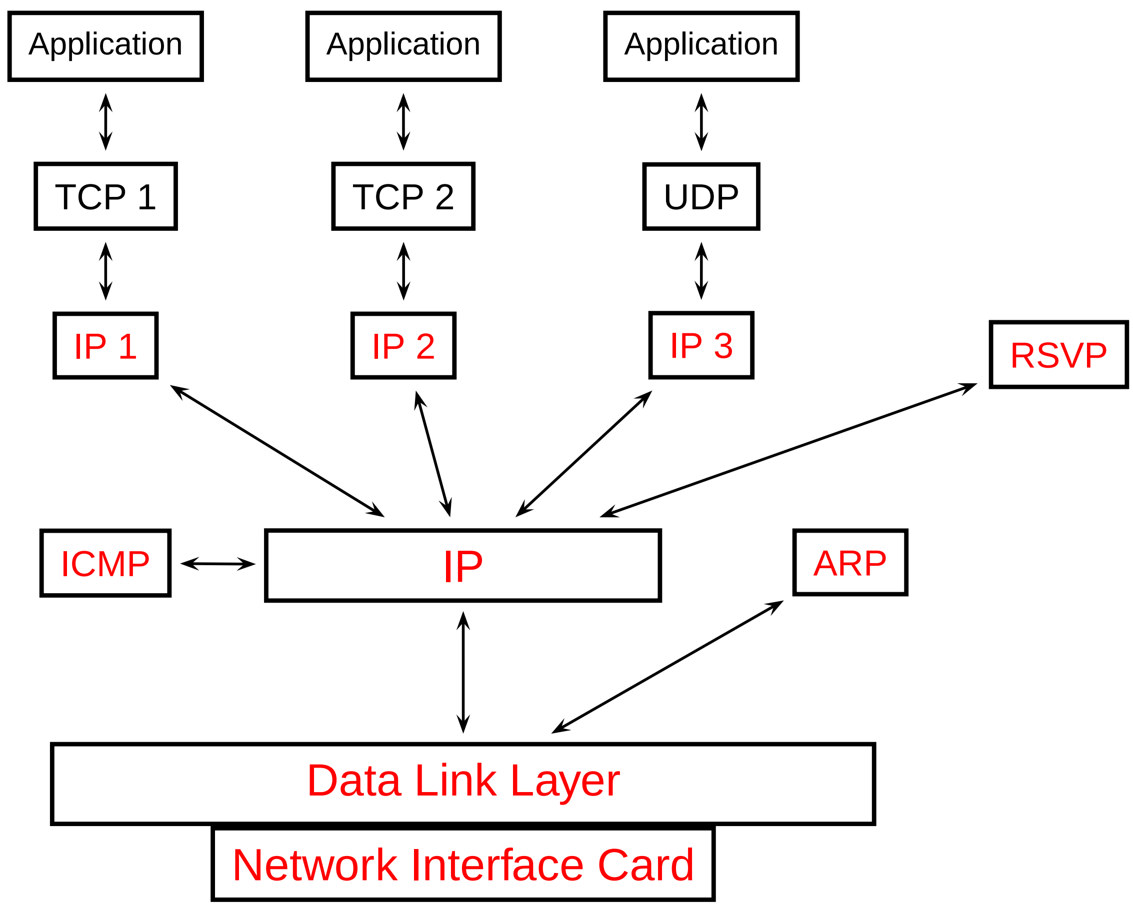
Network layer

The Network Layer ("layer 3") is made of IP and a large number of auxiliary protocols. Some of the auxiliary protocols do not fit very well into the OSI stack structure.



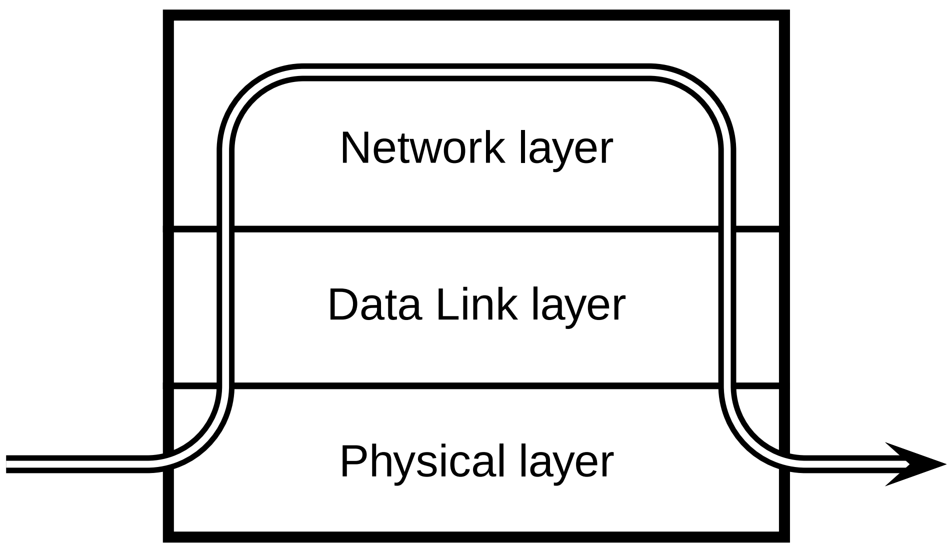
Network layer

The IETF does not deal with the implementation of protocols (nor did OSI). therefore several options exist; they are required to be functionally equivalent.



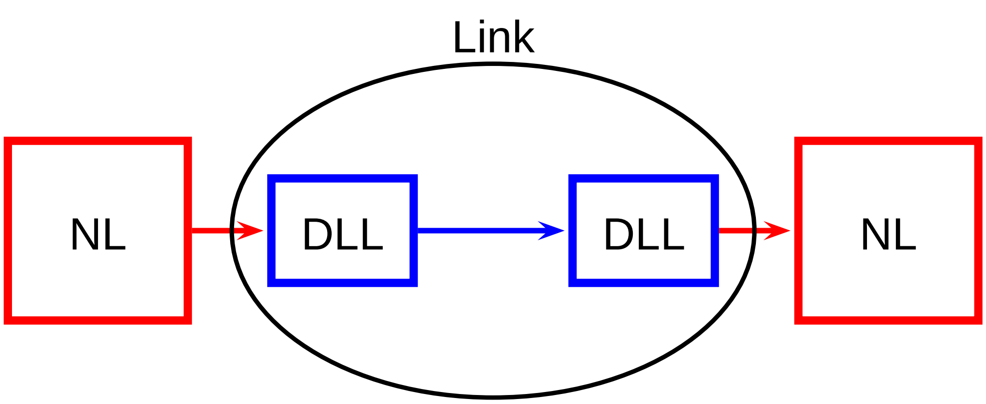
Network router stack

The box which is controlled by NL software is called a **router**.



Network Layer

The NL software controls the flow of packets ("datagrams") in the network core. It does not, however, move packets around; its role is to provide "end–to–end" service (like TL) where the ends are two routers (endpoints of a link).[a]



[a] In TL the ends are hosts.

The Network Layer is responsible for:

- Path determination (routing algorithm).
- Switching (forwarding packets).
- Addressing and handling conventions.
- Fragmentation and reassembly of T–PDUs.
- If needed, call setup (virtual link–to–link circuits within the network cloud).

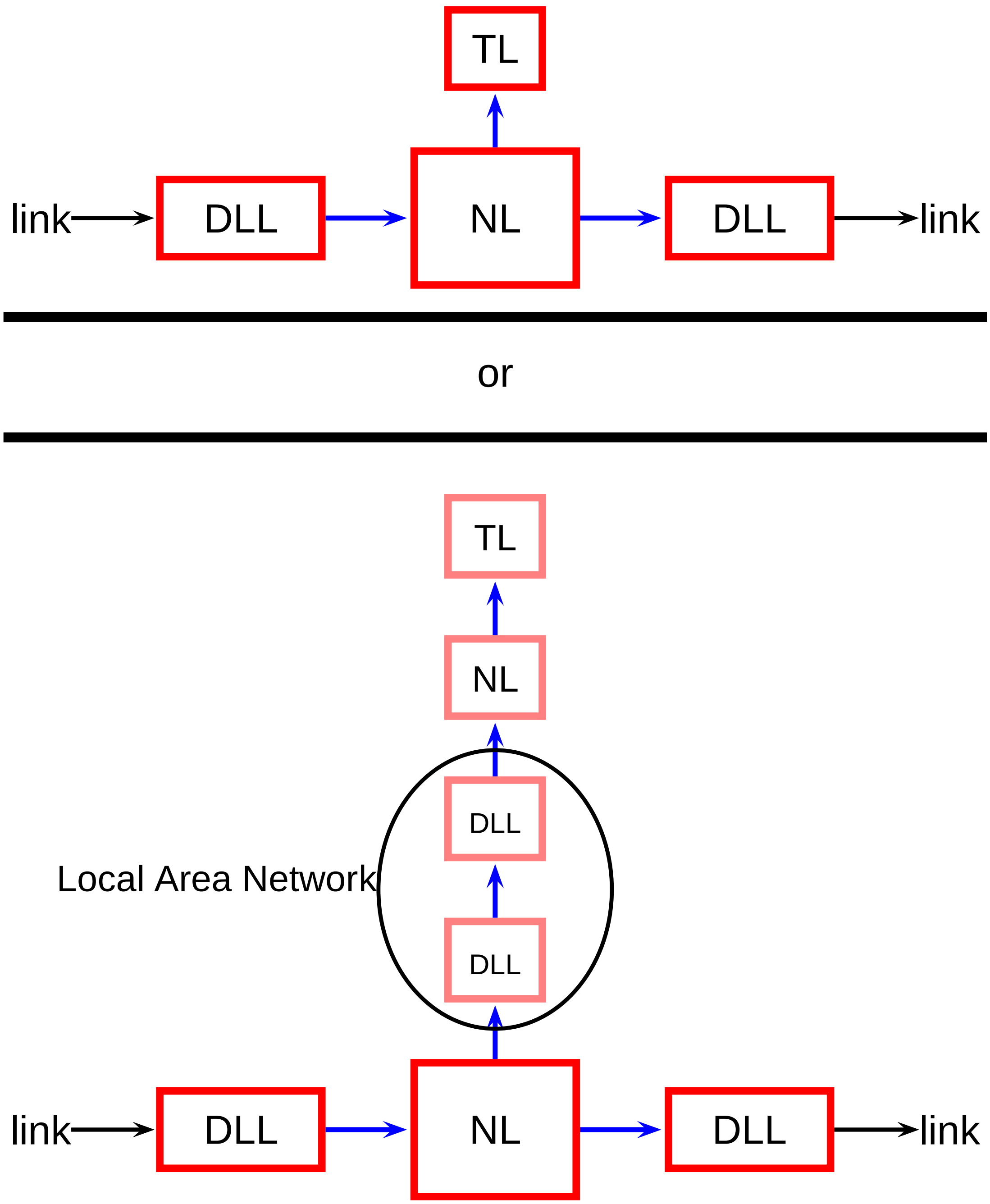
Call setup is done only in some networks; the Internet protocol (IP) does not do it.

Routing vs. forwarding

The life of a packet is divided into 3 phases:

1. Its life in the source host (possibly including the travel to the host's access point).
2. Its journey through the network core. This journey is made of **hops**; before each hop the packet is **routed** and **forwarded**.
3. Its life in destination host (possibly including the trip from the destination host's access point to the host itself).

The first and third phases are deterministic. They involve TL activity in the host and (possibly) DLL activity when transferring the packet between the host and its access point (through a LAN). The NL takes care of the second (and most important) phase.


TL
link
DLL
NL
DLL
link
or
TL
NL
DLL
Local Area Network
DLL
link
DLL
NL
DLL
link

Packet in a router

When a (packet) datagram reaches a router, it is placed in an input buffer. The NL protocol determines the fate of this packet, which can be one of the 3 possibilities:

1. Drop it (the packet disappears without a trace).
2. Pass it to the TL of the destination host.
3. Send it to another router.

If option 3 is chosen the router must pick the next router (this is called **routing**) and move the packet from the input buffer to the appropriate output buffer[a] (**forwarding**).

Many authors use the term “forwarding” to describe the whole physical process of moving the bits of a packet from the input link to the output link.

[a]One input and one output buffer is associated with each link connected to the router.

Network Layer in the Internet

While there are many internets (and many internetworking protocols), the most important is **the** Internet and its main networking protocol, **IP** (Internet Protocol) defined in RFC791.

The main property of **IP** is that it is a connectionless service, essentially complementing **UDP** inside the network core.

IP cannot operate alone anymore[a] and it is complemented by a number of auxiliary protocols.

The auxiliaries are not universally deployed and a router must be prepared to interact with other routers that use different auxiliaries.

[a] Too many routers in a dynamically changing configuration plus ever more stringent demands on timeliness.

The role of IP

IP is responsible for the following:

1. Next–hop determination (part of the routing process).
2. Switching (forwarding packets).
3. Addressing and handling conventions.
4. Fragmentation and reassembly of T–PDUs.

Instead of determining the whole path of a datagram, the **IP** in a router chooses one of the router's neighbours as the destination of the next hop[a] based on its current knowledge of the state of the Internet.

The standard does not impose any rules on selecting the next hop and every router is free to use its own approach; these approaches range from the most primitive "hot potato" algorithm to complex semi–global optimisation algorithms.

---

[a] Using local as opposed to global optimisation in the lingo of algorithm design.

IPv4

IP describes (among other things):

- Addressing conventions: host address formats and how multicast is represented.
- Datagram format.
- Packet handling conventions.

Several other protocols complement IP by regulating how addresses can be assigned, how non–standard addresses should be represented, etc.

Addressing conventions

An **IPv4** address is made of two parts: a **prefix** and a **suffix**. The prefix indicates the “network” and the suffix the “host” within this network (a simplistic view).

Network prefixes must be unique. They are coordinated by IANA, a subset of the *Internet Corporation for Assigned Names and Numbers* (formerly *Internet Assigned Number Authority*). Within each network, it is up to the *Internet Service Provider* to define the meaning of the suffix.

Addresses are further categorised in several ways:

- The address hierarchy beyond the prefix–suffix, hierarchical systems (AS).
- “Classful” vs. “Classless” address blocks.
- “Normal” vs. special addresses, such as multicast, broadcast, loopback, etc.
- Host addresses vs. addresses of whole networks.

IP Address structure

An **IPv4** address is made of 4 bytes and the most popular ("dotted") notation (also known as classful) represents each byte separately:

131.104.48.133

Each byte takes values from 0 (all bits equal 0) to 255 (all bits equal 1).

The exact interpretation of the address depends on the class of addresses that it belongs.

At the NL, addresses are typically filtered using a **network mask**, which looks like an IP address itself. The standard operation is to compare a masked IP address (ip_1) with another address (ip_2) taken from a table:

if(ip_1 & mask == ip_2)

A mask of $255.255.255.255$ implies that all the bits are relevant, a mask of $127.0.0.0$ drops the first bit and the last 24 bits.

IP Address classes

IP (IPv4) divides hosts into 5 classes:

A containing networks 0 through 126.

0	8	31
0	prefix	suffix

B containing networks 128 through 191.

0	1	16	31
1	0	prefix	suffix

C containing networks 192 through 223.

0	1	2	24	31
1	1	0	prefix	suffix

Defunct classes D and E were made of networks 224–255.

Division of address space

Class	Prefix	# networks	Suffix	# hosts
A	7	128	24	16,777,216
B	14	16,384	16	65,536
C	21	2,097,152	8	256

Use ip–address to find what your current **IP** address is (no need to pay anything).

Visit database by country to find out where an **IP** address is from.

Special addresses

Multicast

0	1	2	3	… 31
1	1	1	0	multicast address

Broadcast

Local broadcast is achieved by sending a datagram to the address *127.127.127.127*.

Directed broadcast (local, but to another net) is achieved by sending a datagram to an address made of the network prefix followed by all–ones, i.e. *prefix.127.127.127* in class A, *prefix.127.127* in class B, or *prefix.127* in class C.

Multiple addresses and subnets

Important: A host has one **IP** address for each link (DLL interface) that it is connected to.

Irrelevant: The set of all the entities (hosts and routers) connected to one link is called a **subnet**. It is easy to extend this concept into a hierarchy: a set of subnets connected to the outside world by one link is a **supernet** and so on. The “subnet” concept is obsolete.

Network address

When the **suffix** part of the address is all zeroes, the address denotes a **network** as opposed to any host/subnet within this network. (This is used by some non–standard limited broadcast techniques).

Example: *131.104.0.0* is the network containing all the machines with addresses of the form: *131.104.x.y*.

Local host

During bootstrap, the IP address of the host is not known locally. To obtain it, the host refers to itself as *0.0.0.0*. A fully booted host refers to itself by its IP address or by the *loopback* address: *127.anything* (by convention: *127.0.0.1*).

CIDR–cider

A network mask allows to cut out parts of IP classful addresses but not to specify non–classful addresses or blocks of addresses.

To allow block addressing IETF released CIDR (rfc1518 and rfc1519) standardising other prefix lengths than the standard 8, 16, and 24.

CIDR addressing can be used for masking, but most commonly, a CIDR address is interpreted as a block address,

CIDR prefix size is signalled by the notation:

a.b.c.d/x

where *x* denotes the length of the prefix. The *a.b.c.d* portion of a CIDR address can be any address belonging to the block.

Class	Default prefix mask	CIDR prefix
A	255.0.0.0	/8
B	255.255.0.0	/16
C	255.255.255.0	/24
classless	255.240.0.0	/12
classless	255.255.126.0	?

The last entry filters out all the addresses that do not match the first 2 bytes or the mask 01111110 for the third byte. A CIDR address that matches this mask could look like: $131.104.48.133/23$.

For example: $131.104.48.0/20$ denotes the block of addresses ranging from $131.104.48.0$ to $131.104.63.255$ (the lowest 12 bits are masked out by the $/20$).

Note that $131.104.49.193/20$ and $131.104.48.133/20$ have exactly the same meaning as $131.104.48.0/20$.

Block assignment

When an organisation is assigned a contiguous block of **IP** addresses, it can use CIDR to advertise its subnet. Typically, the **network address** is used.

For example, if a block of 32 addresses from $131.104.49.224$ to $131.104.49.255$ is assigned, the block can be advertised as $131.104.49.224/28$ and $131.104.49.224$ will be considered to be the network address of the block.

When nonstandard addresses are involved, IP must choose the address with the **longest matching prefix** in case of conflicts.

For example: A packet is addressed to *200.23.19.57* and the routing algorithm finds in its routing table two matching IP addresses: *200.23.16.0/20* and *200.23.18.0/23*. Both of them match, but the second matches 23 bits while the first matches only 20. Therefore, the packet must be routed to *200.23.18.0/23*.