IP Addressing and Subnetting

IPv4 and IPv6 addresses are hierarchical identifiers used at the network layer. An IPv4 address is 32 bits long, typically written as four decimal octets (e.g. 192.168.1.1). Classful IPv4 originally defined Classes A, B, C with “natural” masks: Class A (0–127.xxx, /8), B (128–191.xxx, /16), C (192–223.xxx, /24). (Classes D/E are reserved for multicast/experimental.) For example, private IPv4 blocks defined by RFC1918 include 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16.

IPv4 Subnetting and CIDR

In classless addressing (CIDR), an address is written as <prefix>/<length> (e.g. 10.0.0.0/24). The suffix length (prefix) denotes the number of network bits. The remaining bits are host bits. For instance, **172.16.100.0/24** has 24 network bits and 8 host bits. Converting to /28 (i.e. 172.16.100.0/28) uses 28 network bits and only 4 host bits. A simple example: 172.16.100.0/24 has network address 172.16.100.0 and broadcast 172.16.100.255 (with hosts .1–.254). In contrast, 172.16.100.0/28 has network 172.16.100.0 and broadcast 172.16.100.15 (hosts .1–.14), giving 16 total addresses (14 usable) in each /28 subnet. Subnetting a /24 into /28s yields 16 subnets of 14 hosts each.

Common subnet sizes and host counts are:

Prefix	**Subnet Mask**	**Total Addresses**	**Usable Hosts**
/30	255.255.255.252	4	2
/29	255.255.255.248	8	6
/28	255.255.255.240	16	14
/27	255.255.255.224	32	30
/26	255.255.255.192	64	62
/25	255.255.255.128	128	126
/24	255.255.255.0	256	254

Configuration Examples

- **Cisco IOS (IPv4/IPv6):** On an interface:
- Router(config)# interface GigabitEthernet0/0
- Router(config-if)# ip address 192.168.10.1 255.255.255.0
- Router(config-if)# ipv6 address 2001:db8:10:1::1/64 eui-64
- Router(config-if)# no shutdown

The eui-64 keyword instructs IOS to form the 64-bit IPv6 host portion from the MAC (per EUI-64 rules). Without eui-64, you would specify the full 128-bit address.

- **Linux (IPv4/IPv6):**
  - sudo ip addr add 192.168.10.2/24 dev eth0 (assigns IPv4 and subnet).
  - sudo ip link set eth0 up (bring interface up).
  - sudo ip -6 addr add 2001:db8:10:1::2/64 dev eth0 (assigns IPv6 /64). (Alternatively, ifconfig eth0 192.168.10.2 netmask 255.255.255.0 up on legacy systems.)

IPv6 Addressing

IPv6 addresses are 128 bits, written in hexadecimal separated by colons (e.g. 2001:0db8:85a3:0000:0000:8a2e:0370:7334). Because of the vast space (2^128 addresses), IPv6 uses fixed-size subnets. **All IPv6 LANs use /64 prefixes** (64-bit network, 64-bit host). A /64 yields 2^64 (~1.84×10^19) possible addresses. For example, RIPE's chart shows that a /64 contains 2^64 addresses. Common IPv6 allocations are /48 to sites (allowing 65,536 /64 subnets) and /64 per LAN. IPv6 has no concept of broadcast; instead, it uses multicast and neighbor discovery.

EUI-64 and Address Generation

IPv6 stateless address autoconfiguration can derive the interface ID from the MAC via EUI-64. The 48-bit MAC is split and FFFE inserted in the middle to make 64 bits, then the 7th bit (U/L bit) is inverted. For example, MAC 12:34:56:78:AB:CD → 1234:56FF:FE78:ABCD (in binary, invert the 7th bit). Cisco IOS supports this with ipv6 address ... eui-64.

DHCPv6 Prefix Delegation

IPv6 can automatically delegate subnets from an ISP. In **DHCPv6 Prefix Delegation (PD)**, a home router requests a prefix (e.g. a /56 or /48) from the ISP's DHCPv6 server. The ISP assigns a routable prefix to the customer, then the router advertises /64s within that prefix via SLAAC or

DHCPv6. PD lets providers automate IPv6 allocation: e.g. a /48 assignment contains 65,536 /64 subnets.

MAC Addressing, ARP, and RARP

A **MAC address** (Media Access Control address) is a 48-bit hardware identifier burned into (or assigned on) an Ethernet NIC. It is usually shown as six hexadecimal bytes (e.g. 00:1A:2B:3C:4D:5E). Each MAC is unique on its local network. Structurally, the first 24 bits are the **OUI** (Organizationally Unique Identifier) assigned by IEEE to the vendor, and the remaining 24 bits identify the interface. The least significant bit of the first byte indicates **unicast (0)** vs **multicast (1)** addressing; the next bit indicates **global (0)** vs **locally administered (1)**.

Address Resolution Protocol (ARP)

ARP operates between Layer-3 (IP) and Layer-2 (Ethernet) on IPv4 LANs. Its job is to map an IP address to a MAC address. When a host needs to send an IPv4 packet on the local subnet, it issues an ARP Request: a broadcast Ethernet frame (destination FF:FF:FF:FF:FF:FF) saying **“Who has IP X? Tell Y”**, where Y is the sender’s IP. Because the target’s MAC is unknown, the request includes the sender’s own MAC and IP, and the target IP (target MAC is left zero). All hosts on the LAN receive the request; the one whose IP matches X replies with an ARP Response: an Ethernet unicast back to the requester, containing X’s MAC address. The requester learns the MAC and caches the IP→MAC mapping in its ARP table (cache).

Key points:

- **ARP Request:** broadcast frame with the sender’s MAC/IP and target IP, target MAC=0000…0000.
- **ARP Reply:** unicast frame from target containing its MAC and IP.

For example, suppose Host A (IP 10.0.0.11, MAC A1) wants Host B (IP 10.0.0.22). Host A broadcasts “Who has 10.0.0.22? Tell 10.0.0.11 (MAC A1)”. Host B responds with its MAC. After ARP, A can send frames with destination MAC of B. ARP entries time out after minutes (clients ≈1 min, routers ≈2–4 hours). Diagnostic commands include show arp on Cisco and arp -n or ip neigh on Linux.

Reverse ARP (RARP)

RARP is a legacy protocol that **did the opposite of ARP**: a host broadcast its MAC to ask for its own IP address. It was used by old diskless workstations with no local storage. In RARP, the host broadcasts “What is my IP if my MAC = 00:11:22:33:44:55?”; a RARP server on the LAN, with a table of MAC→IP assignments, replies with the matching IP. Once the host receives the RARP reply, it configures that IP on its interface. RARP packets use the same Ethernet type (0x8035) as ARP but with opcode 3 (request) and 4 (reply). RARP is obsolete (replaced by BOOTP/DHCP).

Figure: A host broadcasts a RARP request (above) with its MAC, and a RARP server replies with the IP (below).

MAC vs. IP in LANs

MAC addresses operate at Layer 2 (Ethernet), IP addresses at Layer 3. On a LAN, frames are delivered by MAC, but routed by IP. A device deciding how to send an IP packet first checks whether the destination IP is on the same subnet. If yes, it ARPs for that host's MAC; if not, it ARPs for the **gateway's** MAC. The Ethernet frame is then sent with source MAC = sender, destination MAC = (host's MAC or gateway's MAC). Thus, ARP "glues" IP and MAC: it ensures that each IPv4 address has a corresponding MAC so that packets can be carried on Ethernet.

Overall, MAC addressing uniquely identifies interfaces on the LAN, while IP addressing identifies hosts/networks. In combination, they enable internetworking: IP routes end-to-end across networks, and on each LAN ARP resolves each IP to a physical MAC for delivery. This layered addressing scheme allows a router to forward IP packets between networks, while switches use MAC tables to forward frames within a LAN.

Sources: Definitions and tables are drawn from networking standards and tutorials, illustrating IPv4/IPv6 addressing, subnetting, and ARP/RARP behavior.