OSI Model vs TCP/IP Model — Complete Student Notes

Below are clear, step-by-step notes covering OSI vs TCP/IP, every layer explained, real-world protocol examples, what happens during communication (encapsulation, addressing), **IP addressing (IPv4/IPv6)**, **subnets & subnetting**, **gateways**, and **NAT**.
I'll use small ASCII diagrams, analogies, and worked examples so you can study and apply these concepts.

---

1) OSI Model vs TCP/IP Model — high level

Quick summary / analogy

- Think of sending a parcel:
    - Application = writing the letter (what you want to say).
    - Transport = putting the letter in an envelope with the recipient desk number (port).
    - Network = addressing the parcel with the street address (IP).
    - Data Link = the building receptionist who routes parcels to the right apartment (MAC).
    - Physical = the truck and road that physically move the parcel (bits on wire).

Models

- **OSI (Open Systems Interconnection)** — 7 layers: *Application, Presentation, Session, Transport, Network, Data Link, Physical*. Conceptual, granular.
- **TCP/IP (Internet) model** — 4 layers: *Application, Transport, Internet, Link*. Practical model used for the Internet; maps to OSI layers.

Mapping (high-level)

- OSI Application + Presentation + Session → TCP/IP **Application**
- OSI Transport → TCP/IP **Transport** (TCP, UDP)
- OSI Network → TCP/IP **Internet** (IP)
- OSI Data Link + Physical → TCP/IP **Link** (Ethernet, Wi-Fi, physical media)

---

2) OSI Layers — each layer, role, examples, what happens

I'll work top → bottom (sender side), then describe how encapsulation wraps data. For each layer: purpose, addressing/IDs used, protocols/examples, and what happens during communication.

Layer 7 — Application (OSI) / Application (TCP/IP)

- **Purpose:** Interface for end-user applications (browsers, mail clients). Prepares data for transport.
- **Addresses/IDs:** Application-level identifiers (URLs, email addresses), not network addresses.
- **Protocols/examples:** HTTP/HTTPS, SMTP, IMAP, POP3, FTP, DNS, SSH, Telnet, SNMP.
- **What happens:** Application creates the message (e.g., HTTP GET). May call DNS to resolve domain → gets IP. Data passed down to Transport with destination port (e.g., 80 for HTTP, 443 for HTTPS).

Layer 6 — Presentation (OSI)

- **Purpose:** Data representation, encryption, compression, serialization.
- **Examples:** TLS/SSL (encryption), MIME (email formats), character encoding (UTF-8).
- **What happens:** Data formatting (e.g., encrypt HTTP to HTTPS), converts to bytes. Presentation output becomes payload for Transport.

Layer 5 — Session (OSI)

- **Purpose:** Manages sessions/controls dialogs between applications (establish, maintain, terminate).
- **Examples:** NetBIOS, RPC, session management performed by higher-level protocols (HTTP keep-alive, TLS handshake).
- **What happens:** Session establishment/teardown (e.g., TLS handshake + TCP session). Keeps track of sessions and restores them if necessary.

Layer 4 — Transport (OSI) / Transport (TCP/IP)

- **Purpose:** Reliable/ unreliable end-to-end delivery, segmentation, flow control, multiplexing via ports.
- **Addresses/IDs: Ports** (16-bit). Source port, destination port.
- **Protocols/examples:** TCP (reliable, ordered), UDP (unreliable, low overhead), SCTP.
- **What happens (TCP example):**
    1. Segment creation: take application data → add TCP header (src port, dst port, seq, ack, flags, checksum).

2. **TCP 3-way handshake** to establish connection (SYN → SYN/ACK → ACK).
3. Segmentation: large data split into segments.
4. Flow control (window), retransmission on loss, order guaranteed.
5. Pass segment to Network (IP).

- **What happens (UDP):** No handshake, no retransmit, used for DNS, VoIP, streaming.

Layer 3 — Network (OSI) / Internet (TCP/IP)

- **Purpose:** Logical addressing, routing between networks.
- **Addresses/IDs: IP addresses** (IPv4 32-bit, IPv6 128-bit).
- **Protocols/examples:** IPv4, IPv6, ICMP (control messages), IGMP.
- **What happens:** Encapsulate transport segment into an IP packet with source/dest IP, TTL, protocol number (TCP/UDP). Routers inspect IP header to forward to next hop. If packet passes multiple networks, IP header TTL decremented each hop; fragmentation may occur if packet > MTU.

Layer 2 — Data Link (OSI) / Link (TCP/IP)

- **Purpose:** Node-to-node (same physical link or broadcast domain) framing, MAC addressing, error detection.
- **Addresses/IDs:** MAC addresses (48-bit typically) for Ethernet; VLAN tags (802.1Q).
- **Protocols/examples:** Ethernet (IEEE 802.3), ARP (address resolution), PPP, 802.11 (Wi-Fi).
- **What happens:** IP packet → wrapped in a frame with source and destination MAC, EtherType, frame check sequence (FCS). Switches forward frames by MAC address. ARP resolves IP → MAC on IPv4; IPv6 uses NDP (Neighbor Discovery Protocol).

Layer 1 — Physical (OSI) / Link (TCP/IP)

- **Purpose:** Physical transmission of raw bits—cables, voltages, optical signals, radio waves.
- **Examples:** Ethernet PHY, fiber optics, coax, Wi-Fi radio PHY, bit timing, connectors (RJ45).
- **What happens:** Frames converted to bits/voltages/waves → transmitted over medium → receiver recovers bits → passes to Data Link.

---

3) Encapsulation: What happens to a message traveling from Host A → Host B

ASCII diagram (simplified):

App data (HTTP GET)

↓ (Application)

[TCP header][HTTP data] ← Transport (segment)

↓ (Transport)

[IP header][TCP segment] ← Network (packet)

↓ (Network)

[Ethernet header][IP packet][FCS] ← Data Link (frame)

↓ (Physical)

bits on medium → wire / fiber / radio

Step-by-step (example: HTTP GET to example.com):

1. **Application:** Browser forms GET /index.html. Needs IP → calls DNS (UDP/TCP) to resolve domain to 93.184.216.34.
2. **Transport (TCP):** Browser requests TCP connection to port 80 (SYN). Three-way handshake completes. Data is segmented and assigned sequence numbers.
3. **Network (IP):** Each segment is wrapped in an IP packet with src IP (client) and dst IP (server). TTL set (e.g., 64).
4. **Data Link (Ethernet):** Host checks whether destination IP in same LAN. If yes, ARP resolves server's MAC. If remote, it finds default gateway MAC using ARP. Add Ethernet header (src MAC, dst MAC).
5. **Physical:** Frame transmitted as electrical signals. Switch receives frame, looks at dst MAC → forwards to correct port. Router receives frame destined for remote network → strips frame, inspects IP → chooses next hop → re-encapsulates in new frame for next link. Repeat until destination.
6. **At server:** Reverse process — frame → IP packet → TCP segment → HTTP data → application processes response.

Key things during transit

- **Routing** decisions at Layer 3 (routers).
- **Switching** decisions at Layer 2 (switches).
- **Address translation** (NAT) may modify IP headers at edges.

- **MTU & fragmentation**: If IP packet > MTU of link, fragmentation may occur; routers can fragment IPv4, but IPv6 routers do not fragment—source does Path MTU Discovery.
- **Checksums & CRCs**: Used to detect errors (Ethernet FCS, IP header checksum in IPv4, TCP/UDP checksums).

---

4) Differences & similarities: OSI vs TCP/IP

Similarities

- Both are layered architectures to isolate responsibilities.
- Corresponding functions: end-user apps, transport reliability, routing, link/physical transmission.

Differences

- **Layer count**: OSI = 7 (theoretical); TCP/IP = 4 (practical Internet).
- **Design vs implementation**: OSI is prescriptive/academic; TCP/IP is protocol-oriented (built around IP/TCP).
- **Grouping:** TCP/IP combines OSI's Application+Presentation+Session into Application.
- **Adoption:** Internet uses TCP/IP model; OSI is used as teaching model and conceptual reference.

---

5) IP Addressing — IPv4 and IPv6

IPv4 — structure, notation, examples

- **Length:** 32 bits (4 octets).
- **Notation:** Dotted decimal: 192.168.1.10 (each octet 0–255).
- **Binary vs decimal:** Each octet is 8 bits.
    - Example: 192.168.1.70 → binary:
        - 192 → 11000000
        - 168 → 10101000
        - 1 → 00000001
        - 70 → 01000110
          So full binary: 11000000.10101000.00000001.01000110

- **Valid IPv4 examples:**
  - 10.0.0.1, 192.168.0.100, 8.8.8.8
- **Invalid IPv4 examples:**
  - 256.0.0.1 (octet > 255)
  - 192.168.1 (missing octet)
  - 192.168.1.300 (octet > 255)

IPv6 — structure, notation, examples

- **Length:** 128 bits (16 bytes).
- **Notation:** 8 groups of 4 hex digits separated by :. Example:
  - Full: 2001:0db8:85a3:0000:0000:8a2e:0370:7334
  - Compressed: 2001:db8:85a3::8a2e:370:7334 (double colon compresses consecutive zeros once).
- **Common prefixes:**
  - ::1 — loopback (equivalent to IPv4 127.0.0.1)
  - fe80::/10 — link-local
  - fc00::/7 — unique local addresses (ULA)
  - 2000::/3 — global unicast (public)
- **No broadcasts** — IPv6 uses multicast and anycast instead.
- **IPv6 example valid:** 2001:db8::1
- **Invalid examples:** 12345::1 (hex group > 0xFFFF), 2001::db8::1 (double :: used twice).

---

6) IPv4: classes, CIDR, and conversions

Old classful system (historical; still useful to understand)

- **Class A:** 0.0.0.0 – 127.255.255.255; default mask /8 (255.0.0.0). Large networks (~16M addresses).
  - Example: 10.0.0.0/8 (private)
- **Class B:** 128.0.0.0 – 191.255.255.255; default mask /16 (255.255.0.0)
  - Example: 172.16.0.0/12 (private range: 172.16.0.0–172.31.255.255)
- **Class C:** 192.0.0.0 – 223.255.255.255; default mask /24 (255.255.255.0)

    - Example: 192.168.0.0/24 (private)
- **Class D:** 224.0.0.0 – 239.255.255.255 (multicast)
- **Class E:** 240.0.0.0 – 255.255.255.255 (experimental)

Classful addressing is mostly historical. Modern Internet uses **CIDR (classless)**.

CIDR (Classless Inter-Domain Routing)

- **Notation:** 192.168.1.0/24 → /24 is prefix length (first 24 bits are network).
- **Why CIDR:** Flexible networks, efficient route aggregation.
- **Subnet mask from prefix:** Convert prefix length to dotted decimal by setting first N bits to 1.
    - Examples:
        - /8 → 255.0.0.0
        - /16 → 255.255.0.0
        - /24 → 255.255.255.0
        - /26 → 255.255.255.192 (last octet: 11000000 = 128+64=192)
        - /22 → 255.255.252.0 (third octet 11111100 = 252)
- **Hosts calculation:** usable_hosts = 2^(32 - prefix) - 2 (subtract network and broadcast), except some special cases:
    - /24 → 2^8 - 2 = 254 usable hosts
    - /30 → 2^2 - 2 = 2 usable hosts (often used for point-to-point)
    - /31 → historically 0 usable, now RFC 3021 allows /31 for point-to-point (2 addresses used as hosts, no broadcast)
    - /32 → single host route (host route)

---

7) Subnets and Subnetting — purpose and calculations

Why subnet?

- **Divide** a large network into smaller networks for:
    - Security (segmentation)
    - Performance (smaller broadcast domains)
    - Address management and organization

- Different policy or VLANs

- **Subnetting**: borrowing host bits to create more networks.

Subnet mask — how to calculate

- Convert prefix to mask:
  - Example /26: 26 ones → 11111111.11111111.11111111.11000000
  - Decimal: 255.255.255.192
- Determine **block size** of the subnet in the octet where subnetting occurs:
  - e.g., /26 affects last octet: 256 - 192 = 64 → subnets are increments of 64 in the last octet.

Determine network and broadcast addresses (method)

1. Convert IP and mask (or prefix) to binary (or use decimal block method).
2. **Network address** = IP AND mask (bitwise).
3. **Broadcast address** = network address + (block_size - 1) for that subnet (or set all host bits to 1).
4. **Usable range** = network + 1 → broadcast − 1 (unless special /31 or /32).

Worked examples — step-by-step

Example A — Split 192.168.1.0/24 into /26 subnets

- /24 has 256 addresses. /26 means 26 network bits → host bits = 6 → 2^6 = 64 addresses per subnet.
- **Usable hosts per /26:** 64 - 2 = 62.
- **Subnet masks:** /26 → 255.255.255.192. Block size = 256 - 192 = 64.
- **Subnets and ranges:**
  1. 192.168.1.0/26 → network 192.168.1.0, broadcast 192.168.1.63, usable 192.168.1.1–192.168.1.62.
  2. 192.168.1.64/26 → network 192.168.1.64, broadcast 192.168.1.127, usable 192.168.1.65–192.168.1.126.
  3. 192.168.1.128/26 → network 192.168.1.128, broadcast 192.168.1.191, usable 129–190.

4. 192.168.1.192/26 → network 192.168.1.192, broadcast 192.168.1.255, usable 193–254.

- **Check membership:** Does 192.168.1.70 belong to which subnet? It's between 192.168.1.65–192.168.1.126 → **yes**, the 192.168.1.64/26 subnet.

Example B — Find network for 10.10.5.77/20

- /20 → mask 255.255.240.0. Block size in third octet = 256 - 240 = 16.
- Third octet boundaries: 0, 16, 32, 48, 64, 80, ...
- Third octet of IP is 5 → falls in block 0–15.
- **Network =** 10.10.0.0
  Broadcast = 10.10.15.255
  Usable hosts = 10.10.0.1 – 10.10.15.254.
- Number addresses = 2^(32-20) = 2^12 = 4096 addresses (usable 4094).

Example C — 192.168.0.0/22 (CIDR aggregation)

- /22 → mask 255.255.252.0. Block size in third octet = 256 - 252 = 4.
- Range: 192.168.0.0 → 192.168.3.255 (1024 addresses, usable 1022).

Example D — Binary AND method (network = IP & mask)

- IP: 192.168.1.70 → binary 11000000.10101000.00000001.01000110
- Mask /26: 11111111.11111111.11111111.11000000
- AND gives: 11000000.10101000.00000001.01000000 → 192.168.1.64 network.

---

8) Gateways — default gateway vs routing

What is a gateway?

- **Gateway**: a device that forwards traffic from one network to another. Usually a router or a router + firewall.
- **Default gateway:** the IP on a local network that hosts send packets to when the destination is on a different network (i.e., not in the same subnet). Usually the router's IP on that LAN.

Default gateway vs routing

- **Default gateway:** local host setting (one entry). If no specific route exists for destination, send to default gateway.

- **Routing:** decisions made by routers using routing tables to reach various networks. Routers exchange routing info (static routes, routing protocols like OSPF, BGP, RIP).

Home network example

- **Topology:** Home PC 192.168.1.10/24 — default gateway 192.168.1.1 (router).
  - PC wants to reach 8.8.8.8: sees outside LAN → sends packet to 192.168.1.1 → router performs NAT and forwards via ISP.

Enterprise network example

- **Multiple VLANs:** VLAN 10 192.168.10.0/24, VLAN 20 192.168.20.0/24.
  - Inter-VLAN routing: L3 switch or router has interfaces (SVIs) 192.168.10.1 and 192.168.20.1.
  - PC on VLAN 10 default gateway 192.168.10.1. Router has routes to other internal networks and to internet via edge router.
  - Routers use routing protocols (OSPF inside enterprise, BGP to connect to ISP) to exchange routes.

---

9) NAT — Network Address Translation

What is NAT and why used?

- **NAT** modifies IP addresses/ports in packet headers as they cross a boundary (usually between private network and public Internet).
- **Why:** IPv4 address scarcity, privacy (hides internal addressing), simple firewall-like behavior.
- **Where:** Typically on edge routers or firewalls at organization/ISP boundary.

Types of NAT

1. **Static NAT (one-to-one)**
   - Maps one internal private IP to one public IP permanently.
   - Use case: host in DMZ that must be reachable from Internet (web server).
   - Example: 192.168.1.10 ⇄ 203.0.113.10 (external hosts can reach 203.0.113.10 which maps to internal 192.168.1.10).
2. **Dynamic NAT (pool)**
   - Maps internal private addresses to a pool of public IPs on a first-come, first-served basis.

    - One internal IP mapped to one public IP while session exists.

3. **PAT (Port Address Translation) / NAT overload**
    - Many internal hosts share a single public IP using different source ports.
    - Most common at home routers. Example: 192.168.1.5:12345 → 203.0.113.10:40000
    - Advantages: conserves public IPs.
    - Disadvantage: inbound unsolicited connections require port forwarding.

How NAT modifies packets (example)

- Internal host: 192.168.1.5:43210 sends packet to 93.184.216.34:80.
- Router's public IP 203.0.113.10.
- With PAT, router rewrites source IP:port to 203.0.113.10:50000 and records mapping in NAT table:
    - [203.0.113.10:50000] ↔ [192.168.1.5:43210]
- External server responds to 203.0.113.10:50000. Router consults NAT table and forwards to 192.168.1.5:43210.
- **Diagram** (simplified):

[Host:192.168.1.5:43210] --(private)--> [Router (NAT)] --(public)--> [Server:93.184.216.34:80]

NAT table entry created: 203.0.113.10:50000 -> 192.168.1.5:43210

NAT caveats

- Breaks end-to-end IP traceability (peers see router's public IP).
- Can complicate protocols that embed IP addresses/ports (FTP active mode, SIP) — requires ALG or SDP adjustments.
- IPv6 aims to avoid NAT by having abundant public addresses and other mechanisms (but NAT still used in some networks).

---

10) Practical Notes, Commands & Tools (cheat sheet)

Useful concepts to remember

- **IP address** = logical address for end-to-end routing. **MAC** = physical link address for L2 switching.
- **ARP** resolves IPv4 → MAC on same LAN. **NDP** does similar for IPv6.

- **Default gateway** is used when destination not in the same subnet.
- **CIDR** /n = number of network bits. Use 2^(32-n) to find number of addresses in IPv4 subnet.
- **Usable hosts** = 2^(32-n) - 2 (except /31 and /32).
- **MTU** = maximum frame size of link (commonly 1500 bytes for Ethernet). IP fragmentation occurs if packet > MTU.

Quick conversion table (common)

- /24 → 255.255.255.0 → 256 addresses → 254 usable
- /25 → 255.255.255.128 → 128 addresses → 126 usable
- /26 → 255.255.255.192 → 64 addresses → 62 usable
- /27 → 255.255.255.224 → 32 addresses → 30 usable
- /28 → 255.255.255.240 → 16 addresses → 14 usable
- /29 → 255.255.255.248 → 8 addresses → 6 usable
- /30 → 255.255.255.252 → 4 addresses → 2 usable
- /32 → 255.255.255.255 → single host

Handy commands (Linux)

- ip addr show — show IPs
- ip route show — show routing table
- arp -n or ip neigh — ARP cache
- ping <ip> — test reachability
- traceroute <ip> or tracert (Windows) — path to host (shows router hops)
- tcpdump -i eth0 host 93.184.216.34 — capture packets (root)
- dig +short example.com — DNS lookup
- netstat -tunlp — open ports and related processes

---

11) Practice problems (with short answers)

1. **Problem:** Given 192.168.10.130/25, what is the network, broadcast, and usable IP range?
   Solution: /25 mask 255.255.255.128, block size 128 → subnets: 192.168.10.0/25 (0–127) and 192.168.10.128/25 (128–255).

    - Network: 192.168.10.128
    - Broadcast: 192.168.10.255
    - Usable: 192.168.10.129 – 192.168.10.254.

2. **Problem:** Convert 172.16.5.200 to binary (octet by octet).
   Solution:
    - 172 → 10101100
    - 16 → 00010000
    - 5 → 00000101
    - 200 → 11001000
      Full: 10101100.00010000.00000101.11001000

3. **Problem:** 10.0.5.1/30 — how many usable hosts?
   Solution: /30 → 4 addresses total → usable 2 hosts (e.g., .1 and .2), network .0, broadcast .3.

---

12) Final tips & analogies to remember

- **Encapsulation = Matryoshka dolls**: each layer wraps the data with its own header (and sometimes trailer).
- **MAC vs IP**: *MAC* is like apartment number inside a building (local delivery). *IP* is like street address (routing between buildings).
- **Port numbers** are like office room numbers inside a building (which application/service).
- **Router vs Switch:** Router = post office that reads street addresses (IP) and chooses routes; Switch = building receptionist who forwards to the correct apartment (MAC).
- **NAT** is like a receptionist who forwards mail from outside to apartment occupants but uses the receptionist's address for return mail.

---

13) Quick Review — One-page cheat list

- Layering: App → Transport → Network → Link → Physical.
- Common protocols per layer:
    - App: HTTP, DNS, SMTP, SSH
    - Transport: TCP (reliable), UDP (fast)

    - Network: IPv4, IPv6, ICMP
    - Link: Ethernet, ARP, 802.11
    - Physical: cables/wireless
- IP formats:
    - IPv4 — 32 bits (dotted decimal)
    - IPv6 — 128 bits (hexadecimal, :: shorthand)
- CIDR: /n = network bits → mask = convert N ones to dotted decimal.
- Subnet math: block size = 256 - mask_octet_value in the octet where subnetting happens.
- NAT types: Static (1:1), Dynamic (pool), PAT (many→1 via ports).

If you want, I can:

- Generate a printable PDF of these notes with diagrams and the practice problems filled out.
- Or give step-by-step walkthroughs of **specific** subnetting problems (I can produce several practice exercises and fully solved answers).

Which of those would you like next — printable PDF or more practice problems + solutions?