NETWORKING

For DevOps Engineers and Cloud Engineers

Chandra Sekhar Ravella

1.NETWORKING FUNDAMENTALS

a. IP Address:

📌 1. What is an IP Address?

An IP (Internet Protocol) address is a unique numerical identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication.

It serves two primary purposes:

- Identification of a host or network interface
- Location addressing (i.e., where to send data)

🧠 2. Types of IP Addressing Protocols

IPv4 – Internet Protocol Version 4:

- Most commonly used
- 32-bit address: four decimal numbers (octets)
- Range: 0.0.0.0 to 255.255.255.255

Example: 192.168.10.25

This means:

```
192 → Octet 1
168 → Octet 2
10  → Octet 3
25  → Octet 4
```

Each octet ranges from 0 to 255 (because 2^8 = 256).

IPv6 – Internet Protocol Version 6:

- Newer protocol, designed due to IPv4 exhaustion
- 128-bit address, written in hexadecimal
- Example: 2001:0db8:85a3:0000:0000:8a2e: 0370:7334

Shortened format: `2001:db8:85a3::8a2e: 370:7334.`

It can support **~340 undecillion** IP addresses.

📦 3. Structure of an IPv4 Address

An IPv4 address has:

- **Network Portion** identifies the network
- **Host Portion** identifies the specific device on that network

The division is defined using a subnet mask or CIDR notation (e.g., /24).

Example:

```
IP Address: 192.168.1.10
Subnet Mask: 255.255.255.0 (/24)
→ Network = 192.168.1.0
→ Host = .10
```

🏷️ 4. IP Address Classes

Class	Range	Default Subnet Mask	# Hosts	Used For
A	1.0.0.0 – 126.255.255.255	255.0.0.0 (/8)	16 million	Large networks
B	128.0.0.0 – 191.255.255.255	255.255.0.0 (/16)	65,534	Medium-sized networks
C	192.0.0.0 – 223.255.255.255	255.255.255.0 (/24)	254	Small networks
D	224.0.0.0 – 239.255.255.255	N/A (Multicast)	-	Multicast
E	240.0.0.0 – 255.255.255.255	N/A (Reserved)	-	Research/Experimental

🌍 5. Public vs Private IP Addresses

🔒 Private IP Addresses

- Not routable on the internet
- Used for internal networking (LANs, VPCs, etc.)
- Must be translated (via NAT) for internet access

Range	Class	Example
10.0.0.0 – 10.255.255.255	A	`10.1.1.5`
172.16.0.0 – 172.31.255.255	B	`172.16.10.100`
192.168.0.0 – 192.168.255.255	C	`192.168.1.20`

🌐 Public IP Addresses

- Routable on the internet
- Must be unique across the globe
- Example: 8.8.8.8 (Google DNS)

🧰 6. Tools to View/Analyze IP Addresses

🖥️ **Linux Commands**:

```bash
ip addr          # Show all IP addresses
ip route         # Show routing table
ifconfig         # Show interfaces (older tool)
hostname -I      # Display IP address
```

b. Subnet

📌 1. What is a Subnet?

A subnet (short for "subnetwork") is a logically segmented portion of a larger network. It helps organize and divide an IP network into smaller, more manageable parts.

Think of it like this:

Network = Big neighborhood
Subnets = Individual blocks or streets within the neighborhood

🎯 2. Why Use Subnets?

- ✅ Improve **network performance** by reducing congestion
- ✅ Enhance **security** by isolating systems
- ✅ Make **IP address management** more efficient
- ✅ Allow **broadcast traffic containment**
- ✅ Enable **logical separation** of teams or applications

🧩 3. How Subnets Work

Each subnet has:

- A network address (e.g., 192.168.1.0)
- A subnet mask or CIDR notation (e.g., /24)
- A range of usable IPs (excluding reserved ones)
- A broadcast address (e.g., 192.168.1.255)

📐 4. Subnet Mask and CIDR

A **subnet mask** tells us which part of an IP address identifies the **network,** and which part identifies the **host**.

Example:

- IP Address: 192.168.1.10
- Subnet Mask: 255.255.255.0 (equivalent to /24)
- Network: 192.168.1.0
- Broadcast: 192.168.1.255
- Usable IPs: 192.168.1.1 – 192.168.1.254 (254 hosts)

🌍 5. Types of Subnets

🔓 Public Subnet

- Routable on the internet
- Typically attached to an **Internet Gateway**
- Used for services like:
  - Web servers
  - Load balancers

🔒 Private Subnet

- Not accessible directly from the internet
- Internet access via **NAT Gateway**
- Used for:
  - Application servers
  - Databases
  - Internal tools

📶 6. Subnets and Routing

- Routing tables determine how traffic flows between subnets and out to the internet.

- Public subnets route traffic to an Internet Gateway (IGW).
- Private subnets route to a NAT Gateway for secure outbound access.
- Inter-subnet communication is controlled using firewalls or security groups.

🔐 7. Subnet Security

- Use **ACLs (Access Control Lists)** or **Security Groups** to restrict traffic.
- Keep sensitive resources in **private subnets**.
- Expose only required services to **public subnets**.
- Use **VPC Peering or VPN** to connect private subnets across networks.

🔧 8. Subnet Management Tools

🖥️ Linux Tools:

```bash
ip route show          # Shows routing table
ipcalc 10.0.1.0/24     # Show subnet details
```

c. CIDR (Classless Inter-Domain Routing)

📌 1. What is CIDR?

CIDR stands for Classless Inter-Domain Routing.
It is a method for allocating IP addresses and IP routing that replaces the old class-based IP addressing system.

Instead of rigid classes like Class A, B, and C, CIDR allows for more flexible and efficient IP address allocation.

📐 2. CIDR Notation

CIDR uses the following format:

```
<IP Address>/<Prefix Length>
```

Example:

```
192.168.10.0/24
```

- 192.168.10.0 → the **network address**
- /24 → the number of bits used for the **network prefix**

The rest of the bits are used for **host addresses**.

🧩 3. Why CIDR is Important

- ✅ Avoids IP address wastage
- ✅ Allows fine-grained control over network size
- ✅ Supports route summarization (fewer entries in routing tables)
- ✅ Essential in cloud networking (AWS, Azure, GCP, etc.)

🧮 4. CIDR Prefix Length vs Subnet Mask

CIDR Notation	Subnet Mask	# of IPs	Usable IPs
/8	255.0.0.0	16,777,216	16,777,214
/16	255.255.0.0	65,536	65,534
/24	255.255.255.0	256	254
/30	255.255.255.252	4	2
/32	255.255.255.255	1	0 (host only)

> ✨ The smaller the prefix length (e.g., `/16`), the larger the network and more hosts it supports.

🔍 5. How CIDR Works

CIDR treats IP addresses as bit strings and allows subnetting without being constrained by classes.

For example:

192.168.0.0/22

This is a supernet combining four /24 networks:

- 192.168.0.0/24
- 192.168.1.0/24
- 192.168.2.0/24
- 192.168.3.0/24

It contains:

- 1024 total IPs
- 1022 usable IPs

🧠 6. Binary Breakdown of CIDR

Let's look at 192.168.1.0/26

- Binary: 11000000.10101000.00000001.00000000
- /26 means the **first 26 bits** are for network:
  - Network: 192.168.1.0
  - Subnet Mask: 255.255.255.192
  - Host Bits: 6 bits → 2^6 = 64 IPs
  - Usable IPs: 62 (excluding network and broadcast)

d. What Are Ports

🧠 Basic Definition

A port is a virtual number used by computers to identify specific processes or services on a device.

When data is sent over the internet or a network, the IP address identifies where it's going, and the port number identifies what service it's going to.

Think of it like:

- IP Address = Street address
- Port = Apartment number

📦 Example

When you visit a website:

```arduino
https://example.com
```

Your request goes to:

- IP Address: (e.g., 93.184.216.34)
- Port: 443 (used for HTTPS)

🔢 Port Numbers Range

Port numbers are **16-bit unsigned integers**, so they range from:

```
0 to 65535
```

They are divided into categories:

Range	Name	Description
0 – 1023	Well-known ports	Reserved for system services (HTTP, FTP, SSH, etc.)
1024 – 49151	Registered ports	For user-registered services and applications
49152 – 65535	Dynamic/Ephemeral	Used for temporary connections (e.g., browser making an outbound request)

🌐 Common Well-Known Ports:

Protocol	Port	Description
HTTP	80	Web (insecure)
HTTPS	443	Secure web (SSL/TLS)
SSH	22	Secure shell access
FTP	21	File Transfer Protocol
DNS	53	Domain Name System
SMTP	25	Email sending
POP3	110	Email retrieval
IMAP	143	Email retrieval (modern)
RDP	3389	Remote Desktop Protocol
MySQL	3306	Database service
PostgreSQL	5432	Database service

🛠️ Check Port Availability (Linux):

```bash
# List open ports
sudo netstat -tuln

# Or with ss (modern replacement)
sudo ss -tuln
```

2. OSI MODEL (Open-Source Interconnection)

a. DNS Resolution (Domain Name System)

✅ Purpose:

DNS translates **human-readable domain names** (like www.google.com) into **IP addresses** (like 142.250.190.132) so computers can locate each other on a network.

🧭 How DNS Resolution Works (Step-by-Step)

Let's say you open a browser and go to www.example.com:

1. **Check Local Cache**
    - OS/browser checks if www.example.com was recently resolved and cached.
2. **Check OS Resolver Cache**
    - If not in browser, OS checks local DNS cache.
3. **Query DNS Resolver (usually your ISP's DNS server)**
    - If not found locally, your computer asks a DNS recursive resolver (like 8.8.8.8 – Google DNS).
4. **Recursive Query Process:**
    - **a. Root DNS Server**: Sends referral to .com TLD server.
    - **b. TLD DNS Server**: Sends referral to example.com authoritative server.
    - **c. Authoritative DNS Server**: Returns actual IP address of www.example.com.
5. **Return IP to Client**
    - IP address is returned to your computer and cached for future use.

6. **Connection Established**
    - Now your browser uses the IP to connect to the web server.

🖼️ **Example Flow:**

```rust
User -> Recursive Resolver -> Root Server -> TLD Server -> Authoritative DNS -> IP Address
```

b. TCP Three-Way Handshake

✅ **Purpose:**

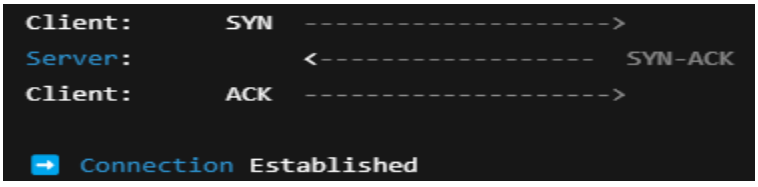
Establish a **reliable connection** between two devices (usually client and server) before actual data is transferred.

🔌 **Steps in TCP Handshake:**

Let's say your browser is connecting to example.com on port 443 (HTTPS):

1. **SYN (Synchronize)**
    - Client → Server: "I want to connect and here's my initial sequence number."
    - TCP Flag: SYN
2. **SYN-ACK (Synchronize + Acknowledge)**
    - Server → Client: "Okay, I acknowledge your request. Here's my sequence number."
    - TCP Flags: SYN + ACK
3. **ACK (Acknowledge)**
    - Client → Server: "Got it. Let's communicate."
    - TCP Flag: ACK

After these 3 steps, a TCP connection is established, and data transfer begins.



c. OSI Model Explained (7 Layers)

Layer	Name	Function	Examples
7	**Application**	User interface, software apps interact here	HTTP, FTP, DNS, SMTP
6	**Presentation**	Data formatting, encryption, compression	SSL/TLS, JPEG, MPEG
5	**Session**	Establishes, maintains, and ends sessions	API calls, RPC, NetBIOS
4	**Transport**	Reliable data delivery, segmentation, flow control	TCP, UDP
3	**Network**	IP addressing, routing	IP, ICMP, IPSec
2	**Data Link**	MAC addressing, error detection (within LAN)	Ethernet, PPP, Switches
1	**Physical**	Physical transmission of bits over medium	Cables, NICs, Hubs, Radio Waves

📶 Layer-by-Layer Breakdown:

🟣 Layer 1: Physical Layer

- **What it does**: Transmits raw bits (0s and 1s) over a physical medium.
- **Examples**: Ethernet cables, fiber optics, Wi-Fi, hubs.
- **Hardware Level**.

🔵 Layer 2: Data Link Layer

- **What it does**: Packages bits into frames, provides error detection, and MAC addressing.
- **Divided into**: Logical Link Control (LLC) and Media Access Control (MAC).
- **Examples**: Switches, Ethernet, Wi-Fi.

🟢 **Layer 3: Network Layer**

- **What it does**: Handles logical addressing and routing (getting data between networks).
- **Adds IP header to packets**.
- **Examples**: Routers, IP, ICMP.

🟡 **Layer 4: Transport Layer**

- **What it does**: Reliable transmission (TCP), or faster/unreliable (UDP). Handles segmentation and reassembly.
- **Examples**: TCP (reliable), UDP (fast), port numbers (22, 443, etc.).

🟠 **Layer 5: Session Layer**

- **What it does**: Manages sessions (start, maintain, end connections).
- **Examples**: NetBIOS, RPC.

🔴 **Layer 6: Presentation Layer**

- **What it does**: Data formatting, encryption/decryption, compression.
- **Examples**: SSL/TLS, ASCII, JPEG, PNG.

🟣 **Layer 7: Application Layer**

- **What it does**: Closest to the user; handles network services like web browsing, email, file transfer.
- **Examples**: HTTP, HTTPS, FTP, SMTP, DNS.

---

ii. 🧭 How OSI Works in Real Life (e.g., Accessing a website)

1. **Layer 7** (Application): You enter a URL in your browser.
2. **Layer 6** (Presentation): SSL encrypts the request.
3. **Layer 5** (Session): A session is established between your browser and the web server.
4. **Layer 4** (Transport): Data is segmented and sent via TCP (port 443).
5. **Layer 3** (Network): IP addresses are used to route the data.
6. **Layer 2** (Data Link): Data is framed and sent over the network via MAC addresses.
7. **Layer 1** (Physical): Bits travel over Ethernet or Wi-Fi.

3. AWS Networking

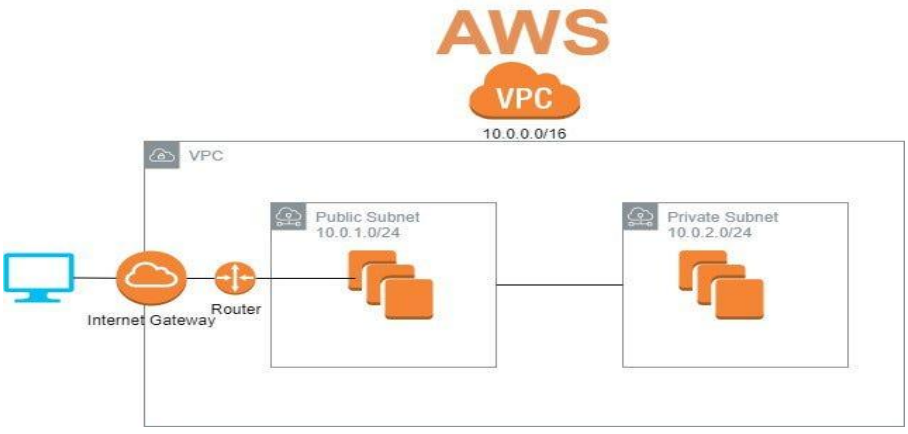
a. Amazon Virtual Private Cloud (VPC)

🔷 What is a VPC?

A **VPC (Virtual Private Cloud)** is a **logically isolated** section of the AWS cloud where you can launch AWS resources (like EC2, RDS, Lambda, etc.) in a **customizable virtual network**.

You have full control over:

- IP address ranges (via CIDR)
- Subnet division (public/private)
- Route tables
- Internet access (via IGW/NAT)
- Firewall rules (Security Groups/NACLs)



📦 Key Components of a VPC

1. CIDR Block (IP Address Range)

- When creating a VPC, you define a CIDR block (e.g., 10.0.0.0/16)
- This determines the range of IPs available for your resources
- Max size is /16 (65,536 IPs), min is /28 (16 IPs)

```bash
CIDR: 10.0.0.0/16 → Usable range: 10.0.0.0 – 10.0.255.255
```

2. Subnets

- **Subnets** divide the VPC into **smaller network segments**.
- Types:
  - 🔓 **Public Subnet**: Has internet access (via Internet Gateway)
  - 🔒 **Private Subnet**: No direct internet access (used for databases, internal services)

  Each subnet must reside in a **single Availability Zone**.

```bash
Example:
Public Subnet → 10.0.1.0/24
Private Subnet → 10.0.2.0/24
```

3. Route Tables

- Define **rules for how traffic is routed** within your VPC and outside it.
- Each subnet must be associated with a route table.

```
Public Route Table:
  - 10.0.0.0/16 → local
  - 0.0.0.0/0 → Internet Gateway

Private Route Table:
  - 10.0.0.0/16 → local
  - 0.0.0.0/0 → NAT Gateway (for outgoing internet)
```

4. Internet Gateway (IGW)

- A **horizontally scaled, redundant** component that **allows internet access** for resources in public subnets.
- Must be **attached to the VPC** and referenced in the route table.

5. NAT Gateway

- Used in **private subnets** to allow **outbound internet access** (for updates, etc.) without exposing the resource to the internet.
- Placed in a **public subnet** with an EIP (Elastic IP).

6. Security Groups (SGs)

- **Stateful firewalls** at the instance level.
- Define **inbound and outbound** rules (e.g., allow SSH, HTTP).
- Return traffic is **automatically allowed**.

7. Network ACLs (NACLs)

- **Stateless firewalls** at the subnet level.
- You must allow both **inbound and outbound** traffic.
- More fine-grained, used for **blacklisting** IPs or ports.

8. DHCP Options Set

- Controls DNS name resolution inside the VPC.
- By default, AWS provides its own DNS resolver (AmazonProvidedDNS).

9. VPC Peering

- Connect two VPCs together **privately** (no internet).
- Great for **cross-account or multi-region** architecture.