**Networking**

Networking is like a highway system that connects cities and states together, from one point to another.

**Computer networking**

Collection of computing devices that are logically connected to communicate and share resources.

* **Node**: like a computer, router, switches, modems, and printers, which are connected through links (a way for data to transmit, such as cables), that follow rules to send and receive data.
* **Host**: is a node that has a unique function. Other devices connect to nodes so they can access data or other services. An example of a host is a server, because a server can provide access to data, run an application, or provide a service.

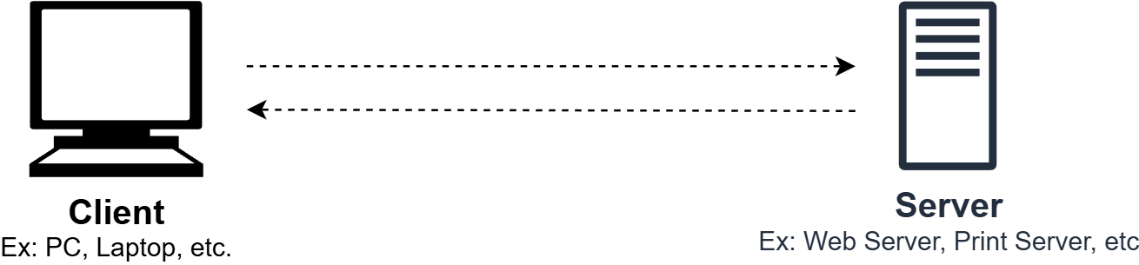
**OSI (Open Systems Interconnection) Model**

It is a standard defining how computers share information over a network.

Mnemonic**: A**aj **P**hir **S**e **T**est **N**hi **D**ena **P**adega

**Client**: is a computer hardware device that accesses the data or a service that is managed by another computer hardware device, which is also called a server.

**Server**: provides a response to a request from a client computer over a network.



**Network Interface Card (NIC)**: Connects a computer to a network, has a unique **MAC address** (Layer 2, OSI model)

**Switch**: connects network nodes and transmits data only to the intended device using **MAC addresses** (Layer 2, OSI model).

Unlike a **hub**, which broadcasts data to all ports, a switch directly links sender and receiver, saving bandwidth. In larger networks, switches create subnets and connect to a router for traffic management.

**Router**: connects multiple networks, directing traffic using **IP addresses** (Layer 3, OSI model).

It filters packets, routes traffic and maintains a route table. In AWS, a route table in a VPC functions like a traditional router, managing network traffic within the cloud.

**Networking Concepts**

**Types of Computer Networks**

**Local Area Network (LAN)**

* Connects devices within a limited area (e.g., office, campus).
* Uses **Ethernet** or **wireless (Wi-Fi)** for high-speed data transfer.
* Example: A router, two switches, and multiple nodes within an office building.

**Wide Area Network (WAN)**

* Connects multiple LANs over a large area (e.g., cities, countries).
* Uses **fibre-optic cables**, **satellites**, and the internet for connectivity.
* Example: Corporate offices in different states connected via WAN (Internet).

Key Difference:

* LAN: Within a building or floor.
* WAN: Connects geographically distant locations.

**Network Topology**

Defines how nodes connect and communicate in a network.

**Types of Network Topologies**

1. **Physical Topology**: Shows the actual layout of devices and cables.
2. **Logical Topology**: Describes how data moves within the network.

**Physical Topologies**

* **Bus**: Devices share a single cable; simple but prone to collisions.
* **Star**: Nodes connect to a central switch; common and efficient.
* **Mesh**: Every node connects to multiple nodes; can be **full** (all connected) or **partial** (some connected).
* **Hybrid**: Mix of topologies (e.g., star-bus); used in large networks.

**Logical Topologies**

* **Bus**: Data moves in one direction, following the physical cable.
* **Star**: Data passes through a central switch before reaching the destination.
* **Mesh**: Fully or partially interconnected nodes ensure redundancy.
* **Hybrid**: Uses multiple logical structures based on network needs.
* **VPC (AWS)**: A **virtual network** in AWS, where users define resources and connectivity.

**Network Management Models**

It representation of how data is managed, and how applications are hosted

|  |  |
| --- | --- |
| Client-Server Model | Peer-to-Peer Model |
| Centralized server manages data and applications. | Each node manages its own data and security. |
| Clients access shared files via the server.  If the server fails, network access is lost. | No central server; devices share files directly. |
| Example: File servers, print servers. | Suitable for small networks with minimal security needs. |

**Network Protocols**

defines rules for data transmission between devices, typically operating at OSI layers 3 (Network) and 4 (Transport).

**Types of Protocols**

* **Connection-Oriented** (TCP): Establishes a session before data transfer (like a phone call). Communication: **synchronous**.
* **Connectionless** (UDP): Sends data without ensuring delivery (like mailing a package). Communication: **asynchronous**.

**Common Protocols**

* **IP (Internet Protocol)**: rules for addressing and routing data across networks. It ensures data reaches the correct destination but doesn’t guarantee delivery.
* **TCP (Transmission Control Protocol)**: Reliable, ordered and error-checked data delivery via a **three-way handshake** (**SYN → SYN/ACK → ACK**). It also gracefully closes communication (like saying goodbye) using **FIN → FIN/ACK → ACK**. If a connection closes abruptly, **RST** (reset) flags signal an error.
* **UDP (User Datagram Protocol)**: Fast, connectionless, and unreliable but efficient for real-time communication.

**Why do TCP and IP work together?**

Think of sending a package via a courier service:

* **IP (address on package)**: it ensures the package reaches the right location.
* **TCP (delivery receipt system)**: it guarantees the package arrives intact, in the correct order, and gets acknowledged upon delivery.

They work together because just like a courier needs both an address (IP) and a tracking system (TCP) for secure delivery, the internet relies on both to ensure reliable data transmission.

**Internet Protocol (IP)**

It is a network protocol establishing rules for relaying and routing data on the internet. It uses IP addresses to identify devices and port numbers to identify endpoints.

**IP Addresses** (Layer 3, OSI Model)

* Unique Identifier: Identifies a device on a network.
* Functions: Identifies **host & network** and used for **location addressing**
* Assignment:
  + **Static**: Fixed, does not change (Ideal for **servers**, **printers**, etc.)
  + **Dynamic**: Changes over time (Used for **laptops**, **mobile devices**)
* Types: **Public** (Internet-accessible) and **Private** (Internal network use only)

**Classes of IP Addressing**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Class | 1st Octet of IP Address | Subnet Mask | Network/Host | Number of networks | Total Addresses |
| A | 1 – 126 | 255.0.0.0 (/8) | N.H.H.H | 126 | (2^24)-2= 16,777,214 |
| B | 128 – 191 | 255.255.0.0 (/16) | N.N.H.H | 16,384 | (2^16)-2=65,534 |
| C | 192 – 223 | 255.255.255.0 (/24) | N.N.N.H | 2,097,152 | 254 |
| D | 224 – 239 |  |  |  |  |
| E | 240 – 254 |  |  |  |  |

Note:

* **0.0.0.0/8** reserved for default route
* **127.0.0.0/8** reserved for loopback address
* **169.254.0.0/16** for link-local address. This range also called **automatic private IP addressing.**

**Private IP Ranges (RFC 1918)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | Network | IP Range | Subnet Mask | Total Addresses |
| A | 10.0.0.0/8 | 10.0.0.0 – 10.255.255.255 | 255.0.0.0 | 2^24= 16,777,216 |
| B | 172.16.0.0/12 | 172.16.0.0 – 172.31.255.255 | 255.240.0.0 | 2^20=1,048,576 |
| C | 192.168.0.0/16 | 192.168.0.0 – 192.168.255.255 | 255.255.0.0 | 2^16=65,536 |

**Public vs. Private IPs**

* Public IP → Accessible over the internet (like 54.239.28.85)
* Private IP → Used within internal networks (e.g., 10.0.0.0)

**IPv4 Addressing**

Format: 32-bit number, written in **dotted decimal** (e.g., 10.15.200.0)

Structure:

* **Network Portion**: Identifies the network.
* **Host Portion**: Identifies individual devices.

**IPv4 Breakdown**

Consists of four octets (0-255), each 8-bit binary.

Example: 10.100.20.5 → 10 (8-bit) | 100 (8-bit) | 20 (8-bit) | 5 (8-bit) = 32-bit

Each bit is **0 or 1**, determining its **decimal value**. Understanding this helps with subnetting and IP allocation.

**IPv6 Addressing**

Next-gen IP standard, extending IPv4’s address space exponentially.

Format: 128-bit **hexadecimal** address, separated by colons (**:**)

(e.g., 2600:1f18:22ba:8c00:ba86:a05e:a5ba:00FF).

**IPv6 Advantages**

* **Larger Address Space**: Supports **340 trillion, trillion, trillion** addresses.
* **Enhanced Security**: Built-in encryption & authentication.
* **Efficient Packet Handling**: Reduces latency & improves performance.

**IPv6 vs. IPv4**

* IPv4 → 32-bit (4.2 billion addresses)
* IPv6 → 128-bit (massive scalability for IoT & future tech)

**Port Numbers (Endpoint)**

While an IP address identifies a **device**, a **port number** is a unique identifier that directs network traffic to the right **application/service** running on that device.

Example:

* IP Address (Hospital) → 192.168.1.1
* Port (Doctor's Office Extension) → Port 22 (SSH)

**Common Port Numbers & Their Uses**

* **Port 22** → SSH (Secure remote login)
* **Port 53** → DNS (Translates domain names to IPs)
* **Port 80** → HTTP (Standard web traffic)
* **Port 443** → HTTPS (Secure web traffic)
* **Port 3389** → RDP (Remote Desktop access)

**How Ports Work in Networking**

* Allows multiple applications to communicate over the same IP
* Helps route messages correctly
* Used in firewall rules & security settings

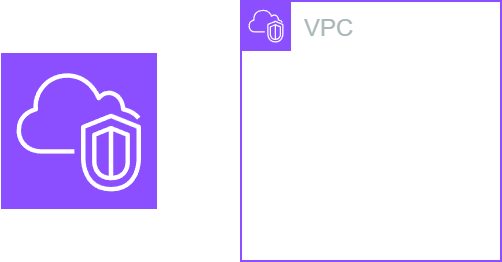
**Networking in the AWS Cloud**

|  |  |
| --- | --- |
| Traditional topology | AWS service |
| Data center | Amazon VPC |
| Router | Route tables |
| Switches (subnets) | Subnets |
| Firewall | Security groups and network access  control lists (network ACLs) |
| Servers and operating systems | Amazon Elastic Compute Cloud (Amazon  EC2) instances |
| Modem | Internet gateway |

**Amazon Virtual Private Cloud (VPC)**

It lets you create a **logically isolated** section of AWS, mimicking a **traditional data center** but with the **flexibility, scalability, and security** of the cloud.

It allows you to **define and control** your virtual network, customize IP addressing, manage traffic flow, and securely deploy AWS resources.



Why Use Amazon VPC?

* **Full Control**: Customize your network's IP range, routing, and security settings.
* **Resembles a Data Center**: Just like a physical data center, it requires subnets, routing, and security configurations but can be deployed **in minutes**.
* **Secure & Scalable**: Offers built-in security controls like **security groups and network ACLs** while scaling resources seamlessly.

**How Amazon VPC Works**

Imagine launching an **EC2 instance (virtual server)** inside a VPC.

You can configure it to be publicly accessible (using an **Internet Gateway**) or keep it private (within a **private subnet**).

**VPC Features**

* Dedicated to **one AWS account**

A white square with black text

AI-generated content may be incorrect.

* Belongs to a **single AWS Region**

A white and blue square with a flag

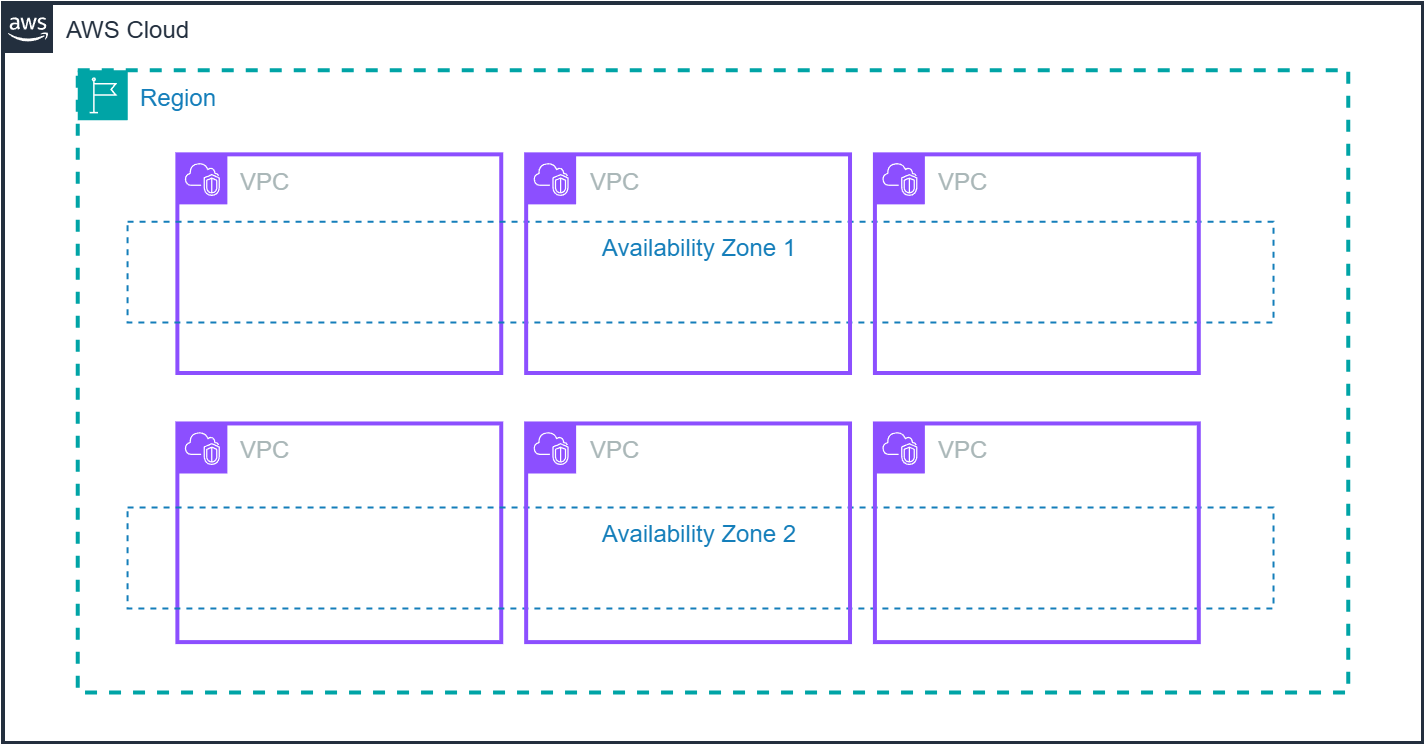
AI-generated content may be incorrect.

* Can span **multiple Availability Zones (AZs)**

A white square with blue squares

AI-generated content may be incorrect.

* Is **logically isolated** from other VPCs



Amazon VPCs can span multiple **Availability Zones** within an AWS Region, allowing for isolated and scalable networking environments. You can create multiple VPCs per AWS account and define subnets within them, though minimizing subnet count simplifies network management.

**IP Addressing in Amazon VPC**

* **CIDR Block**: smallest /28 (16 IPs), largest /16 (65,536 IPs).
* **Private IPs**: Use **RFC 1918** ranges (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16).
* **No Overlapping**: Ensure non-conflicting IPs when connecting networks.
* **Fixed Range**: CIDR cannot be changed post-creation but **secondary CIDR** can be added.

**AWS CLI Command to Create a VPC**

aws ec2 create-vpc --cidr-block 10.0.0.0/16

**AWS Reserved IPs:**

Within each subnet CIDR block, AWS reserves the **first four** and **last** IP addresses:

* 10.0.0.0 – Network address
* 10.0.0.1 – VPC router
* 10.0.0.2 – DNS server
* 10.0.0.3 – Reserved (future use)
* 10.0.0.255 – Broadcast (not supported in VPC)

**Public IPs**: Assigned via Elastic IPs; not recommended outside private ranges.

**Key Components of VPC**

* **CIDR Block**: Defines the IP range (e.g., **/16 – /28** ) for your VPC.
* **Subnets**: Divide the VPC into smaller segments, allowing better network organization.
* **Route Table**: Controls traffic flow between subnets and external networks.
* **Internet Gateway**: Enables internet access for resources within the VPC.
* **VPC Endpoint**: Connects AWS services privately without using the public internet.

**Subnet**: A network within a network, dividing a large network into smaller interconnected networks.

Components:

* **Network ID**: Uniquely identifies the subnet.
* **Subnet Mask**: Defines IP range and separates network/host bits.
* **Host ID Range**: Usable IPs between subnet and broadcast address.
* **Usable Hosts**: Varies by class and prefix.
* **Broadcast ID**: Sends data to all hosts in the subnet.

**Subnets Characteristic:**

* **Subnet**: **range of IP addresses** within a VPC
* **Availability Zones (AZs)**: subnet cannot span multiple Availability Zones; **one subnet per Availability Zone**
* **Public Subnet**: Routes traffic to an **Internet Gateway (IGW)** via an associated **route table**
* **Private Subnet**: Traffic **is not routed** to the internet.
* **Subnet Sizing**: CIDR blocks of multiple subnets within **a VPC cannot overlap**.
* **Subnet Limits**: upto **200** subnets per VPC.

**Security Group**: Firewall rules for instances (stateful).

* Controls inbound/outbound traffic.
* Default group assigned if none specified.
* Associated with instances.

**Internet Gateway**

* Enables communication between **VPC and the internet**.
* Horizontally scaled, redundant, and highly available.
* Essential for internet access from the VPC.

**Public Subnet & Routing**

* Public subnet’s route table must have **0.0.0.0/0** → **IGW-xxxxx**.
* Instance must have a public IPv4 or Elastic IP to communicate with the internet.

**Steps to Enable Internet Access in AWS Console**

1. **Create & attach** the **Internet Gateway (IGW)** to the VPC.
2. Update the subnet’s route table → **0.0.0.0/0 → IGW**.
3. Ensure instances have a public IPv4 or Elastic IP.
4. **Verify security groups & network ACLs** allow inbound/outbound traffic.

**NAT Gateway**

* Enables private subnet instances to connect outside the VPC.
* Prevents external entities from initiating connections (RESET flag sent).
* **Public Subnet**: NAT Gateway assigned an Elastic IP (public IP) and placed in a public subnet.
* **Private Subnet**: Route Table: 0.0.0.0/0 → nat-xxxxx (NAT Gateway); No need for public IP on private subnet instances.

**Route Table**

* Holds **routes** and **targets** to direct network traffic within a VPC.
* **Destination**: IP address or CIDR range (e.g., **0.0.0.0/0** for internet access).
* **Target**: **gateway** or **network interface** for routing destined traffic.
* **Route Table Association**: Each **route table** must be associated with a subnet

Key Considerations in AWS Console:

* **Public route tables** should be associated with **public subnets** for internet access.
* **Destinations** hold **IP addresses/ranges**, while **targets** hold **services**
* A **route table with an Internet Gateway (IGW)** enables **public subnet** access.

Troubleshooting Importance

* Incorrect **associations** can break **internet/network connectivity**
* A subnet can be associated with **only one route table**, but a **route table** can be linked to **multiple subnets.**

**Elastic Network Interface (ENI)**

A **virtual network interface (NIC)** attached to an **EC2 instance.**

Acts as a **connection point** between the instance and the network.

Each ENI has:

* Primary IPv4 address (from the VPC range)
* Optional secondary IPs
* MAC address
* Security groups

**Primary vs. Secondary ENIs**

**Primary ENI**:

* **Created by default** for each instance.
* **Cannot be detached** from the instance.

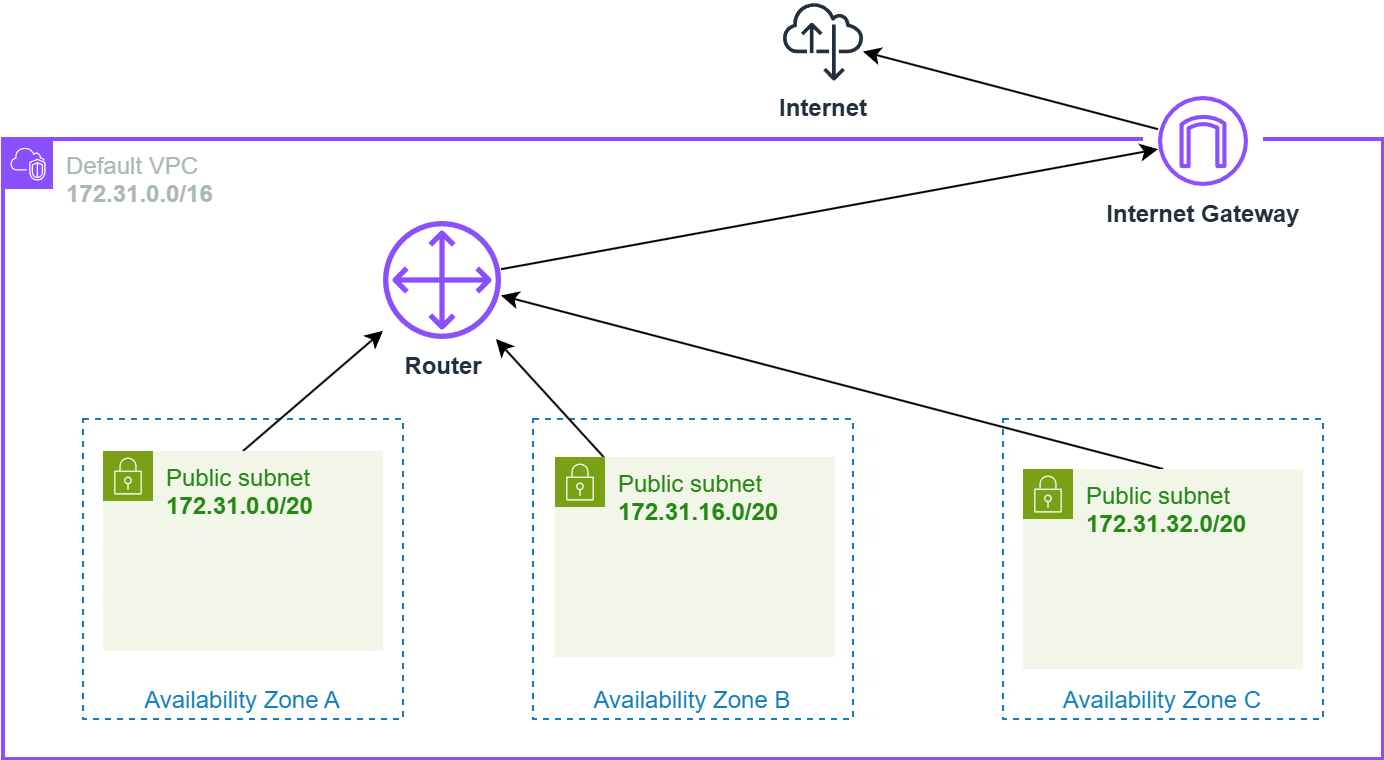
**Secondary ENIs**

* Can be **added**, **detached**, and **moved** between instances.
* Keeps its **IP address & attributes** when moved.
* Redirects traffic to the **new instance** when attached elsewhere.

**Use Cases for Multiple ENIs**

* **Network segregation**: e.g., public & private interfaces.
* **High availability**: failover between interfaces.
* **Security isolation**: assigning different security groups per interface.
* **Traffic management:** dedicated ENIs for specific workloads.

**Finding Your IP Address**: Use the **ipconfig** command in the Windows command prompt to find IP addresses, subnet masks, and other network information.

**Default VPC**

|  |  |
| --- | --- |
| Destination | Target |
| 172.31.0.0/16 | local |
| 0.0.0.0/0 | internet\_gateway\_id |

**What is a DNS Server?**

A **Domain Name System (DNS) server** converts **hostnames** (e.g., **www.example.com**) into **IP addresses** (e.g., **192.0.2.1**).

**AWS DNS Setup**

* **Amazon Route 53 Resolver** is the default **DNS server** for AWS VPCs.
* It resolves **internal domain names** within the VPC.
* Performs **recursive lookups** for **external domains** using public name servers.

**Custom DNS Options**

If the default Route 53 Resolver is not suitable, you can:

* **Use a custom DNS server** (configured via **DHCP options**).
* **Use an Amazon Route 53 private hosted zone** for internal DNS resolution.

**Key Network Protocols & Diagnostic Tools**

* **HTTP (Hypertext Transfer Protocol)**: Accesses web pages via URLs.
* **HTTPS (Hypertext Transfer Protocol Secure)**: Secure web communication, crucial for IoT.
* **ICMP (Internet Control Message Protocol)**: Diagnoses network issues (e.g., `ping`).
* **DHCP (Dynamic Host Configuration Protocol)**: Automatically assigns IP addresses.
* **DNS (Domain Name System)**: Resolves domain names to IP addresses.
* **Telnet**: Checks if a remote port is open.

**Diagnostic Tools**

* **hping3**: Packet generator & security testing.
* **traceroute**: Tracks packet path to a destination.
* **mtr**: Combines **ping** & **traceroute** for real-time analysis.
* **nslookup**: Queries DNS records.

**Additional Networking Technologies**

**Wireless Technologies**:Devices communicating wirelessly without traditional cords.

Examples:

* **WEP (Wired Equivalent Privacy)**: Early encryption method (40-bit key, insecure).
* **WPA (Wi-Fi Protected Access)**: Improved encryption (256-bit key).
* **Bluetooth Low Energy (BLE)**: Low-power connectivity for IoT, healthcare, fitness, security.
* **5G Cellular Systems**: High-speed, low latency (up to 10 Gbps).

**Evolution**: Constant improvements for security, efficiency, and reliability.

**Internet of Things (IoT)**: Network of physical devices collecting and sharing data.

Primary Goal: Real-time self-reporting for efficiency and automation.

**Key Features**:

* Transfers data without human interaction.
* Expands product capabilities.
* Generates and analyses data.

Examples:

* Consumer: Smartphones, wearables, connected cars, thermostats.
* Enterprise: Medicine pumps, smart fleet management, CCTV cameras, smart cities.

**Communication Process**:

* **Devices**: Sensors & actuators collect data.
* **Protocols**: Lightweight communication (e.g., MQTT).
* **Gateways**: Send data to the cloud for processing.
* **Processing**: Rule-based actions, AI analysis, alerts to users.
* **User Interface**: Mobile apps, web interfaces for monitoring and control.

**AWS IoT Core**

|  |  |
| --- | --- |
|  | **Cloud-based IoT Platform** for managing IoT devices. |

**Key Functions**: Data management, analytics, security, application support.

**Communication**: Uses secure protocols (MQTT, HTTPS) for IoT device integration.

**Amazon WorkSpaces**

|  |  |
| --- | --- |
| A white line drawing of a cube  AI-generated content may be incorrect. | Managed Desktop-as-a-Service (DaaS) solution by AWS. |

**Functionality**:

* Provides cloud-based virtual desktops.
* Enables secure remote access to applications and data.
* Supports Windows and Linux environments.

**Use Cases**:

* Secure remote work.
* Application development and testing.
* High-performance computing needs.

**Benefits**:

* Scalable, secure, and cost-effective.
* Reduces IT overhead with managed infrastructure.
* Provides persistent, high-performance desktop experience.