# Tutorial on Socket Programming and More

Parsa Pazhooheshy
CSC 458- Winter 2025

# Outline

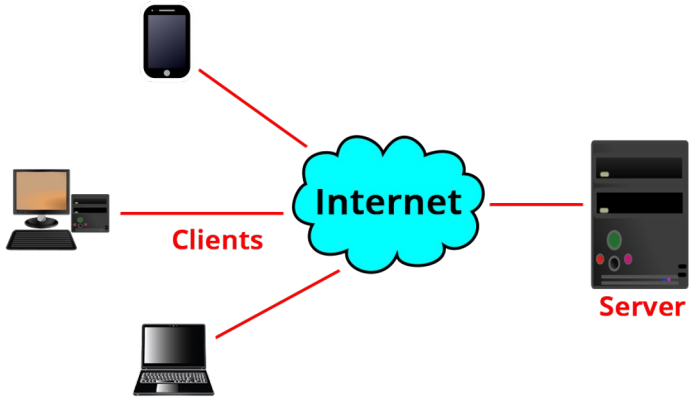
- This tutorial is a gentle introduction to some different concepts that you will see through the course.
- You will learn more about these concepts throughout the course. This tutorial is supposed to be just a starting point.

# Internet as a Black Box

**Definition**

- The Internet is a vast, globally interconnected network that enables communication between devices (clients and servers) using standardized protocols.

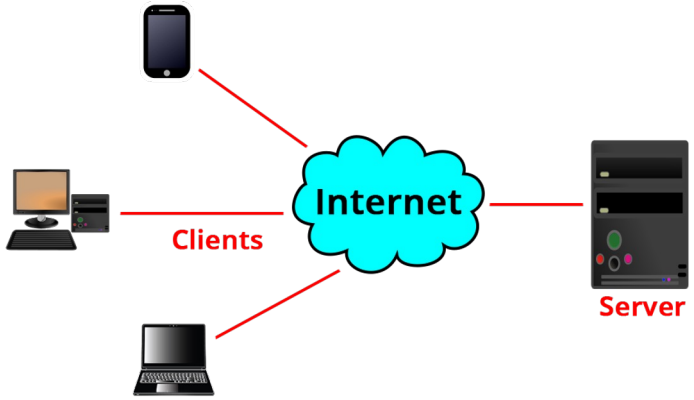
- Facilitates seamless data exchange between clients and servers over a publicly accessible infrastructure.



# Internet as a Black Box

## Characteristics

- Scalability & Decentralization: No single entity owns or controls the entire Internet; it is built on a distributed and scalable architecture.
- Packet-Switched Communication: Data is transmitted in small packets over multiple paths, improving efficiency and reliability.

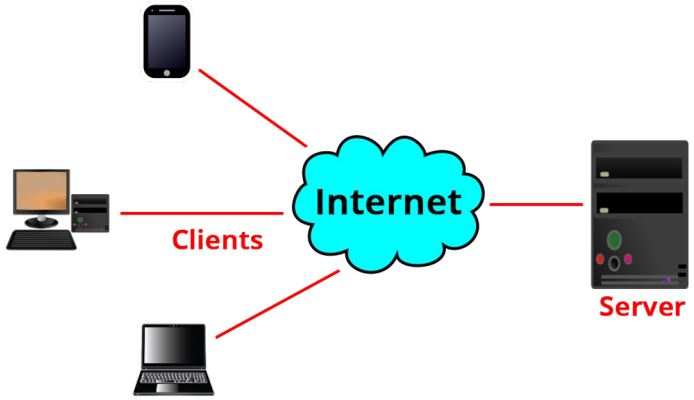


# Internet still as a Black-Box!

**Devices**

- Clients: Devices (computers, smartphones, IoT devices) that request and consume data or services.

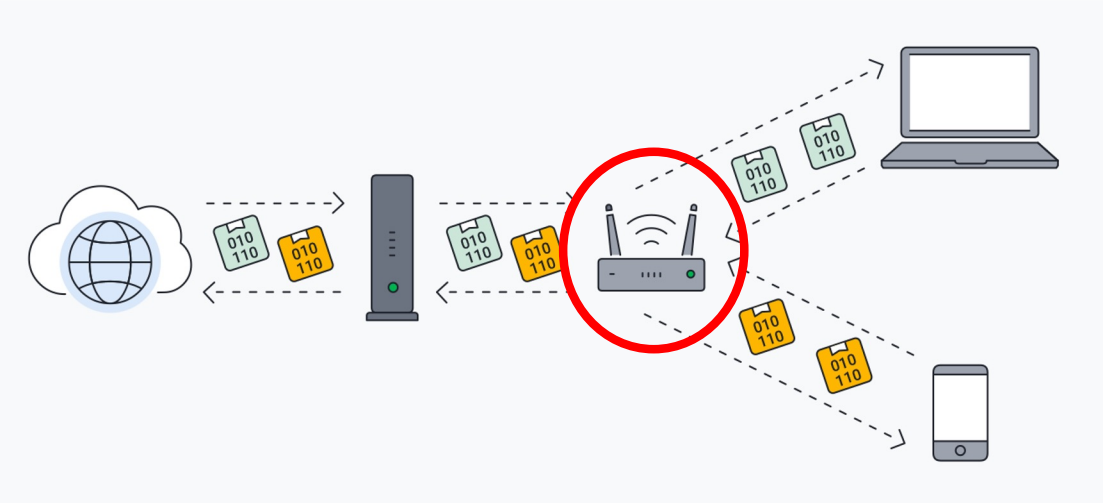
- Servers: Systems that provide resources, data, and services (e.g., web servers, cloud storage).



# Internet not as a Black-Box any more!

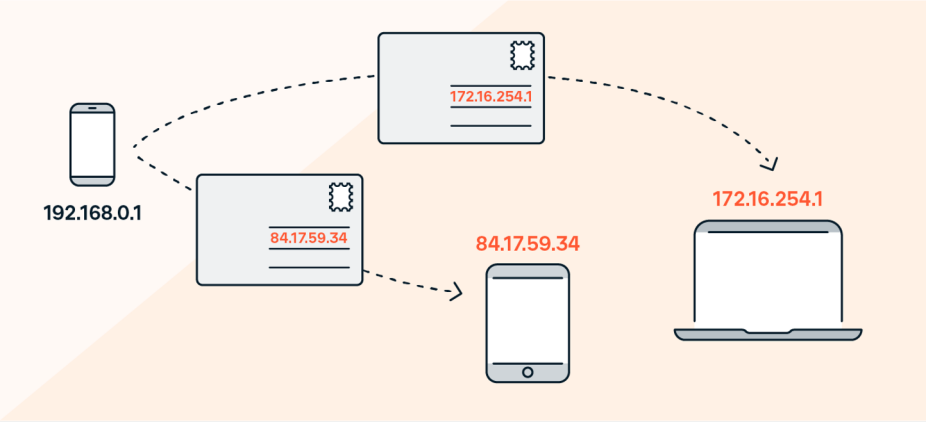
## Devices

- Routers & Switches: Direct network traffic and ensure data packets reach their destination efficiently.



# IP Address: The Internet’s Addressing System

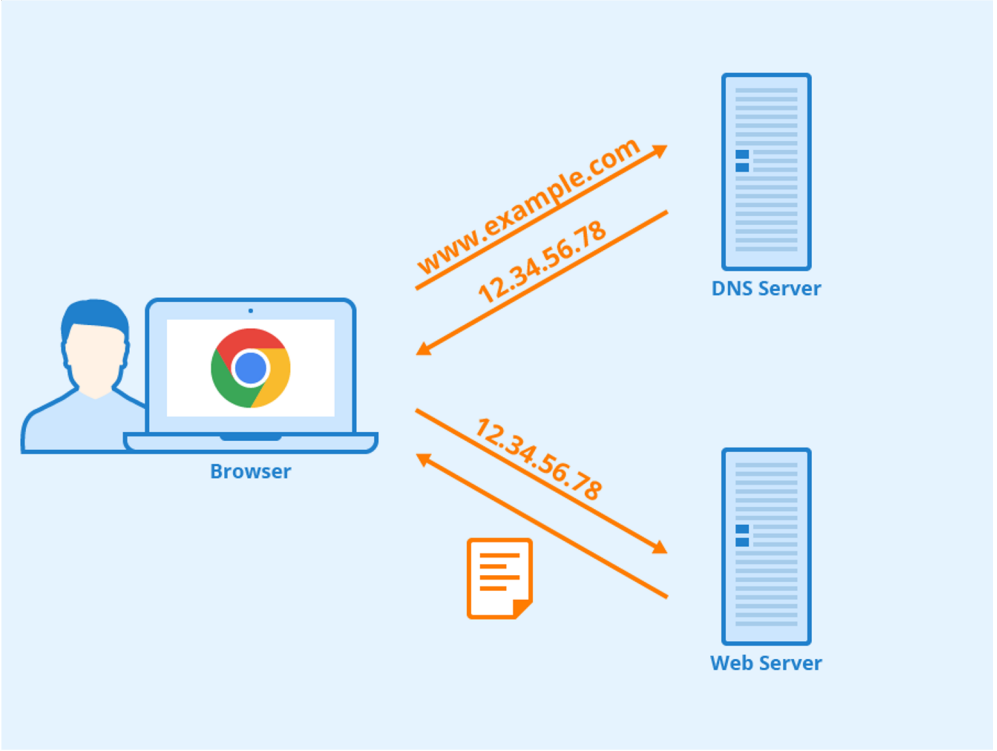
- A unique numerical identifier assigned to each device on a network.
- Used for device identification and communication across the Internet.
- Similar to a home address but for digital devices.



# Internet not as a Black-Box any more!

## Devices

- Domain Name System (DNS):
  - When a user types a domain name into a browser, the browser sends a DNS query.
  - A DNS server responds to the query by providing the IP address for the domain.
  - The browser uses the IP address to communicate with the website's server.



# Step by Step Process of a Web Request

1. You type www.example.com in your browser.
2. The DNS translates www.example.com to an IP address (e.g., 192.168.1.1).
3. Your device sends a request to the web server hosting the website.
4. The server processes the request and sends back the HTML, CSS, and JavaScript needed to display the website.
5. Your browser renders the website, and you can interact with it.

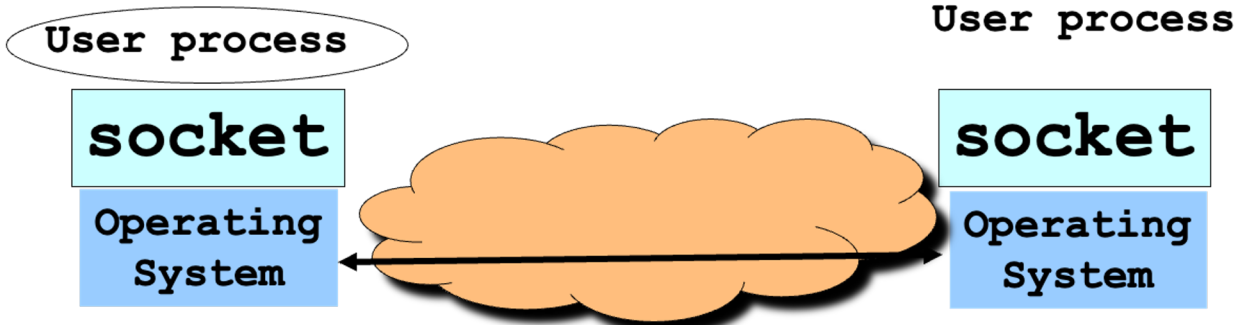
# Recap

What we have covered so far

1. Internet as a black box
2. Servers and Clients in the Internet
3. Routers and Switches
4. DNS

# Network Socket

- Network sockets are application-level software implementation that enable communication between two processes over a network (which can be Internet or any other network type).
- It acts as a bridge between applications and the network stack, allowing processes to send and receive data.
- The API for the network protocol stack creates a handle for each socket created by an application, commonly referred to as a socket descriptor.



| Function | Purpose |
| --- | --- |
| socket() | Creates a new socket. |
| bind() | Assigns an IP address & port to the socket. |
| listen() | Sets up a socket to accept connections (server-side). |
| accept() | Accepts an incoming connection request. |
| connect() | Establishes a connection to a remote socket. |
| send()/recv() | Sends or receives data over a socket. |
| close() | Closes the socket after communication.[12] |

# Typical Client Program

**Prepare to communicate**

Create a socket

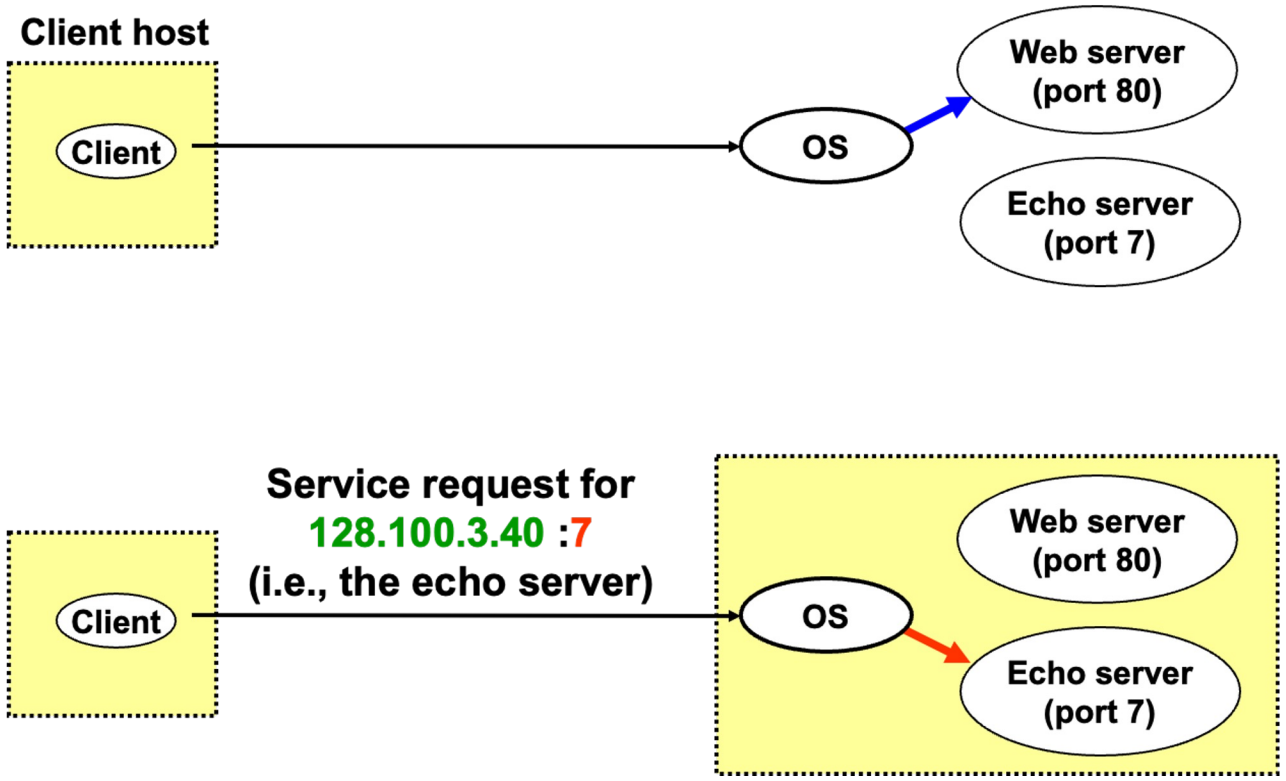
- Determine server **address** and **port number**
- Initiate the connection to the server

Exchange data with the server

- Write data to the socket
- Read data from the socket
- Do stuff with the data (e.g., render a Web page)

Close the socket

# Using Ports to Identify Services



# Socket Parameters

- A socket connection has 5 general parameters:
  a. The protocol
     i. – Example: TCP and UDP
  b. The local and remote address
     i. Example: 128.100.3.40
  c. The local and remote port number
     i. Some ports are reserved (e.g., 80 for HTTP)
     ii. Root access require to listen on port numbers below 1024

# Servers Differ From Clients

## Passive open

- Prepare to accept connections
- … but don’t actually establish one
- … until hearing from a client

## Hearing from multiple clients

- Allow a backlog of waiting clients
- ... in case several try to start a connection at once

## Create a socket for each client

- Upon accepting a new client
- … create a new socket for the communication

# Typical Server Program

Prepare to communicate

- Create a socket
- Associate local address and port with the socket

Wait to hear from a client (passive open)

- Indicate how many clients-in-waiting to permit
- Accept an incoming connection from a client

Exchange data with the client over new socket

- Receive data from the socket
- Send data to the socket
- Close the socket

Repeat with the next connection request