Application Layer - Additional Clarity Keywords

Study-Ready Notes

Compiled by Andrew Photinakis

October 17, 2025

Contents

1 Ports

1.1 What is a Port?

1. A port is like a doorway into a computer for network communication.
2. Every device on a network (like your laptop, a web server, or a phone) has:
   - A unique IP address, which identifies the device.
   - Multiple ports, which identify specific applications (processes) running on that device.
3. Analogy: Think of the IP address as the street address of an apartment building, and the port numbers as the apartment numbers inside it.

1.2 Why we need ports?

1. Many programs can use the network at once:
   - Your web browser.
   - Your email client.
   - A game you are playing online.
   - A video call using Zoom or Skype.
2. All of these share the same IP address (your computer's address on the network), but each communicates using a different port number so the OS knows which process should get each incoming packet.
3. Without ports, your computer would not know whether an incoming packet was meant for your browser or your email app.

1.3 Structure: IP + Port = Socket

1. Each network connection is identified by a socket pair:

   ```
   (source IP, source port) → (destination IP, destination port)
   ```

2. Example (you visiting a website):
   - `(192.168.1.10, 49523) → (128.119.245.12, 80)`
   - Your computer (client) uses a temporary port like 49523 assigned by your OS.
   - The web server listens on port 80 for HTTP requests.

1.4 Port Number Ranges

Range	Description	Examples
0–1023	Well-known ports (assigned by IANA)	HTTP 80, HTTPS 443, SMTP 25, DNS 53
1024–49151	Registered ports (specific apps)	MySQL 3306, NFS 2049
49152–65535	Dynamic/private ports (temporary)	Client ephemeral ports

1.5 Common Port Numbers

Service	Protocol	Port
HTTP	TCP	80
HTTPS (Secure HTTP)	TCP	443
FTP (File Transfer)	TCP	21
SMTP (Email sending)	TCP	25
IMAP (Email retrieval)	TCP	143
DNS	UDP/TCP	53
SSH (Secure Shell)	TCP	22
Telnet	TCP	23

1.6 Ports and Protocols (TCP vs UDP)

Feature	TCP Port	UDP Port
Connection-based	Yes	No
Reliability	Guaranteed	Best effort
Use case	Web, Email, File Transfer	DNS, Streaming, Gaming

1.7 Ports and Security

1. Ports can act as entry points for attacks.
2. Firewalls are configured to block or allow specific ports.
3. Port scanning tools (like `nmap`) are used to identify open or vulnerable ports.
4. Example: A secure web server only opens port 443 (HTTPS) instead of 80, ensuring all traffic is encrypted.

1.8 Real Example: Visiting a Website

1. Your browser creates a TCP connection.
2. The OS assigns an ephemeral (temporary) port on your machine, e.g., 54321.
3. It sends a packet to:

```
destination IP: 128.119.245.12
destination port: 80
```

4. The server receives it on port 80, where its web server software (like Apache) is listening.
5. The server replies from (80) → (54321).
6. When done, the connection closes and your port 54321 becomes free again.

1.9 Summary

Concept	Explanation
Port	A number identifying a process or service on a host.
IP + Port	Together identify a specific communication endpoint.
Server port	Fixed, well-known (e.g., 80, 443).
Client port	Temporary, dynamically assigned.
Socket pair	Defines one full connection between two hosts.
Firewall use	Controls access to ports for security.

1.10 Quick Recap

- Ports separate traffic for multiple applications on one device.
- Port numbers range from 0–65535.
- Well-known ports are reserved for common services.
- Clients use ephemeral ports; servers use fixed ones.
- Both TCP and UDP use port numbers, but handle connections differently.

2 Ports in Computer Networks

2.1 Introduction

In computer networks, a **port** is a logical endpoint for communication that allows multiple networked applications to coexist on a single device. While an **IP address** identifies *which device* on the network to reach, a **port number** specifies *which process or service* within that device should receive the data. This combination of IP address and port number forms a **socket** — the foundation of process-to-process communication across the Internet.

[Summary: Ports serve as numbered entry points that allow multiple applications to share the same network connection on a device.]

2.2 Keyword Breakdown

- **Port Number:** A 16-bit integer (0–65535) identifying a specific application or process.
- **Socket:** The pairing of an IP address and port number (e.g., 192.168.1.10:443).
- **Well-Known Ports:** Reserved for standard Internet services (0–1023).
- **Registered Ports:** Assigned by IANA to specific applications (1024–49151).
- **Dynamic or Private Ports:** Used temporarily by client applications (49152–65535).

[Mnemonic: "W-R-D" — Well-known, Registered, Dynamic — helps recall the three port ranges.]

2.3 Analogy: Apartments in a Building

An IP address is like the *street address* of an apartment building, and ports are the *apartment numbers*. Data arriving at the building (IP address) must know which apartment (port) to go to. For instance, a web server might live in apartment 80, while an email server lives in apartment 25.

[Summary: The port number directs network messages to the correct application, much like an apartment number directs mail within a building.]

2.4 How Ports Function in Communication

Every Internet connection involves two endpoints, each identified by a socket:

Socket Pair: (Source IP, Source Port, Destination IP, Destination Port)

When a client (browser) requests a webpage:

1. The browser selects a random **source port** (e.g., 51820).
2. The web server listens on a known **destination port** (e.g., 443 for HTTPS).
3. The request is sent as:

$$192.168.1.5 : 51820 \rightarrow 93.184.216.34 : 443$$

4. The server's response is sent back along the same path, using the port numbers to ensure data reaches the right application.

[Summary: Ports ensure that each networked process on a host receives the correct data among multiple concurrent communications.]

2.5 Port Number Categories

[Summary: Port numbers are divided into standardized ranges to manage global consistency and avoid conflicts.]

Range	Type	Example Usage
0–1023	Well-Known Ports	HTTP (80), HTTPS (443), SMTP (25)
1024–49151	Registered Ports	MySQL (3306), PostgreSQL (5432)
49152–65535	Dynamic / Private Ports	Temporary client connections

Table 1: Port number categories and examples.

2.6 TCP vs. UDP Ports

Both TCP and UDP protocols use port numbers, but for different purposes:

Protocol	Type	Example Port	Description
TCP	Connection-Oriented	80 (HTTP)	Reliable data delivery via acknowledgment
UDP	Connectionless	53 (DNS)	Faster transmission with no delivery guarantee

Table 2: Comparison of TCP and UDP port usage.

[Summary: TCP ports manage reliable, ordered communication; UDP ports handle faster, simpler message delivery.]

2.7 Common Port Numbers in Practice

Service	Protocol	Port
HTTP	TCP	80
HTTPS	TCP	443
FTP	TCP	21
SSH	TCP	22
DNS	UDP/TCP	53
SMTP	TCP	25
POP3	TCP	110
IMAP	TCP	143
DHCP	UDP	67, 68

Table 3: Commonly used port numbers in networking.

[Mnemonic: "2-1-2-2-5-3-8-6" pattern for FTP (21), SSH (22), SMTP (25), DNS (53), POP3 (110), IMAP (143), HTTPS (443) helps recall key ports.]

2.8 Visual Diagram Description

Imagine a diagram showing:

- A client on the left labeled with "Source IP: 192.168.1.5, Port: 51820".

- A server on the right labeled "Destination IP: 93.184.216.34, Port: 443 (HTTPS)".
- Arrows between them representing TCP segments or UDP datagrams.

This visualization helps reinforce how port numbers map communication between specific processes on each host.

[Summary: Visualizing sockets clarifies how each connection is identified by IP and port on both ends.]

2.9 Integration with the Broader Network System

Ports operate within the **Transport Layer** (Layer 4 of the OSI model) but are essential to the **Application Layer** (Layer 7) where specific network services reside. The transport layer (e.g., TCP/UDP) uses port numbers to multiplex multiple application streams across a single IP connection.

[Summary: Ports bridge the gap between transport mechanisms and application processes, enabling simultaneous communication across many services.]

2.10 Key Takeaways

[Summary: Ports are numerical identifiers that distinguish network services on a host, allowing simultaneous communication over shared IP connections. They are crucial for process-to-process data delivery in TCP/IP networks.]

[Mnemonic: "IP = device, Port = program" — IP locates the machine, Port locates the process.]

3 Ports in Computer Networks

3.1 Core Definitions

- **Port:** Numerical identifier used to direct network traffic to the correct application or process.
- **Socket:** Combination of IP address + port number; uniquely identifies a communication endpoint.
- **Well-Known Ports:** Ports 0–1023 reserved for standard services (e.g., HTTP=80, HTTPS=443).
- **Ephemeral Ports:** Temporary ports (1024–65535) assigned by client for short-lived sessions.

3.2 Keyword Breakdown

- **IP Address:** Device location on network.
- **Port Number:** Specific application/service identifier on a device.
- **TCP vs UDP:**
  - TCP: Connection-oriented, reliable.
  - UDP: Connectionless, faster, no delivery guarantee.
- **Multiplexing:** Multiple applications share one IP using different ports.

3.3 Step-by-Step Function

1. Client chooses ephemeral port & sends request to server IP + well-known port.
2. Server receives packet, inspects destination port.
3. Packet routed to correct application/service.
4. Response sent back to client socket (IP + ephemeral port).
5. Communication continues until session ends.

3.4 Examples & Applications

- HTTP: Port 80 → Web pages
- HTTPS: Port 443 → Secure web pages
- FTP: Ports 20/21 → File transfers
- SSH: Port 22 → Secure remote login
- DNS: Port 53 → Domain name resolution

3.5 Comparisons / Contrasts

- **Port vs IP:** IP = device, Port = application on device.
- **TCP vs UDP Ports:**
  - TCP: Reliable, ordered, connection-based.
  - UDP: Fast, unordered, connectionless.
- Well-Known vs Ephemeral:
  - Well-Known: Fixed for standard services.
  - Ephemeral: Dynamic, temporary for clients.

3.6 Analogies

- IP Address = House address
- Port = Specific room in the house
- Socket = House + Room (full delivery location)

3.7 Visual / Diagram Description

- Server = building, rooms = ports
- Client sends packets → addressed to room number (port)
- Responses follow reverse path to client's ephemeral port

3.8 Concept Integration

- Works at Transport Layer (OSI Layer 4)
- Allows multiple applications to share a single IP
- Essential for TCP/IP networking and client-server models

3.9 Summary & Study Aids

[Summary: Ports identify applications on a host, enabling organized network communication with IP addresses and sockets.]

[Mnemonic: IP = House, Port = Room, Socket = House + Room]