

# **Networks**

Tópicos de Informática para Automação

---

Mário Antunes

November 03, 2025

Universidade de Aveiro

# Why Networking Matters i

Networking is the invisible fabric of the modern world. It's no longer just about computers; it's about *everything*.

- **Communication:** From email and social media to video calls.
- **Services:** Cloud computing, streaming (Netflix, Spotify), and online gaming.
- **Economy:** E-commerce, banking, and global financial systems.
- **IoT (Internet of Things):** Smart homes, wearable tech, and connected cars.

Understanding networking is no longer optional; it's a fundamental skill for any technologist.

# Key Network Equipment i

First, let's meet the hardware that builds a network.

- **Hub:** A “dumb” repeater. Any packet it receives is broadcast to *every* other port. It’s slow, inefficient, and creates “collisions.” (Rarely used today).
- **Switch:** A “smart” device for a LAN. It learns which device is on which port (using MAC addresses) and sends packets *only* to the intended recipient.
- **Router:** A “gateway” that connects *different* networks. Your home router connects your private LAN to your provider’s WAN (the Internet).

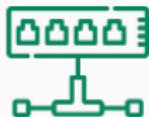
## Key Network Equipment ii

---

- **Access Point (AP):** A “translator” that connects wireless devices (using Wi-Fi) to the wired network (the switch).
- **ONT (Optical Network Terminal):** Your “modem” for a fiber-optic connection. It translates light signals from the fiber cable into electrical signals for your router (Ethernet).

# Key Network Equipment iii

## Common Types of Network Devices



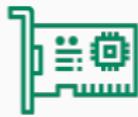
Hub



Router



Gateway



NIC



Modem



Repeater



WAP



Firewall



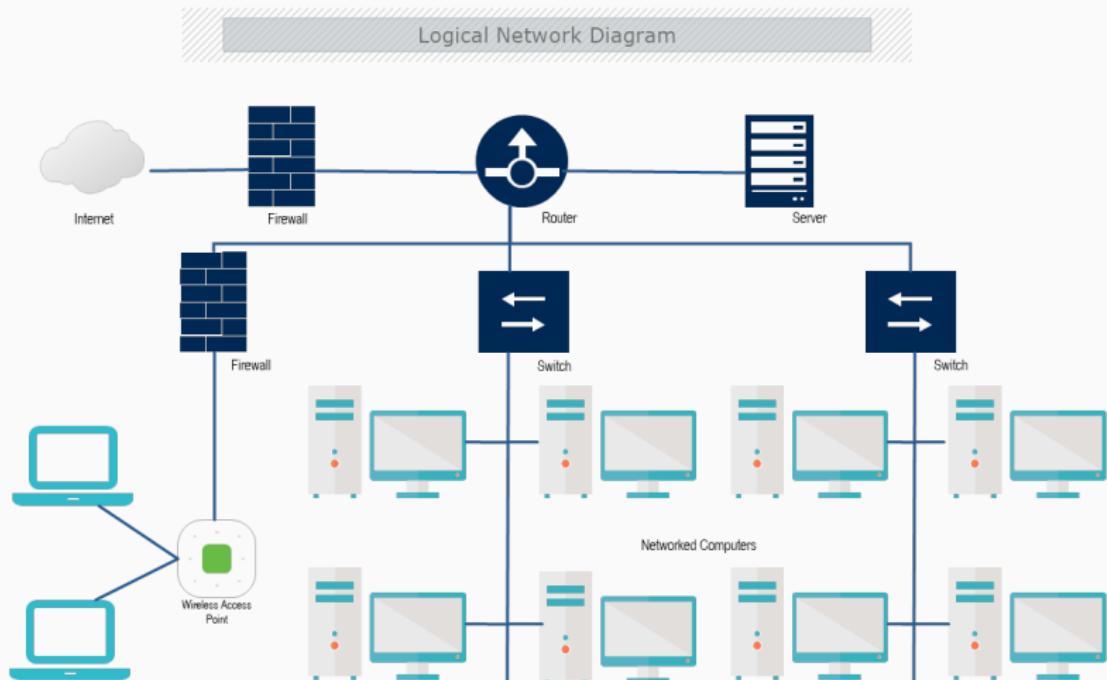
IDPS



VPN

**Figure 1:** Common Network Devices

# Key Network Equipment Connected



**Figure 2:** Network Devices Connected

# The Two-Address System: MAC & IP

Every device on a network has **two** addresses. Both are crucial.

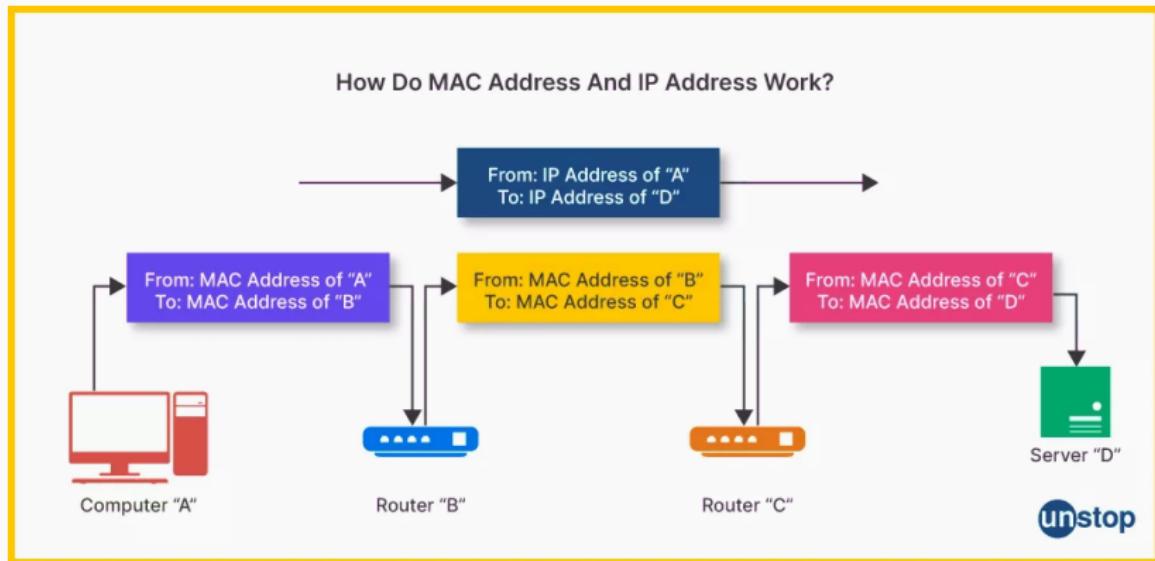
- **MAC Address (Physical Address):**
  - Example: 00:1A:2B:3C:4D:5E
  - A unique, 48-bit serial number burned into the network card by the manufacturer. It is permanent.
  - **Used for:** Communication *within* the same Local Area Network (LAN).
- **IP Address (Logical Address):**
  - Example: 192.168.1.10

## The Two-Address System: MAC & IP ii

- A logical, 32-bit (or 128-bit for IPv6) address assigned to the device by the network (e.g., by your router). It is temporary.
- **Used for:** Communication *between* different networks (on the WAN/Internet).

**Analogy:** A MAC address is like your **passport number** (permanent, identifies you). An IP address is like your **home address** (logical, changes if you move).

# The Two-Address System: MAC & IP iii



**Figure 3:** From MAC to IP

## How LANs *Really* Work: ARP i

Your computer (192.168.1.10) wants to send a packet to your printer (192.168.1.15) on the same LAN.

- The **Router** only understands IP addresses.
- The **Switch** (which connects them) only understands MAC addresses.

How does the computer find the printer's MAC address?

1. It shouts to the whole LAN: "WHO HAS 192.168.1.15?"  
This broadcast is the **Address Resolution Protocol (ARP)**.
2. The printer (192.168.1.15) replies: "I DO! My MAC is 00:AB:CD:EF:12:34."

## How LANs *Really* Work: ARP ii

---

3. Your computer stores this IP → MAC pair in its **ARP table** and sends the packet.

# The Language: IPv4 & The Subnet Mask

An IPv4 address alone isn't enough. It's always paired with a **Subnet Mask**.

- **IP Address:** 192.168.1.10
- **Subnet Mask:** 255.255.255.0

The subnet mask's job is to split the IP into two parts:

1. **Network Part:** 192.168.1.x ("What street am I on?")
2. **Host Part:** x.x.x.10 ("What is my house number?")

This is how your computer knows if another IP is **local** (on the same network) or **remote** (on a different network).

# Special IPv4 Address Ranges

Not all IPs are equal. They are reserved for specific uses.

- **Loopback Address (localhost):**

- 127.0.0.1
- This address always means “**this computer.**” It’s a virtual interface used for testing applications on your own machine.

- **Private / LAN Addresses:**

- 10.0.0.0 – 10.255.255.255
- 172.16.0.0 – 172.31.255.255
- 192.168.0.0 – 192.168.255.255
- These are for use *inside* a private network (LAN). They are not routable on the public internet.

## Special IPv4 Address Ranges ii

- **Public / WAN Addresses:**

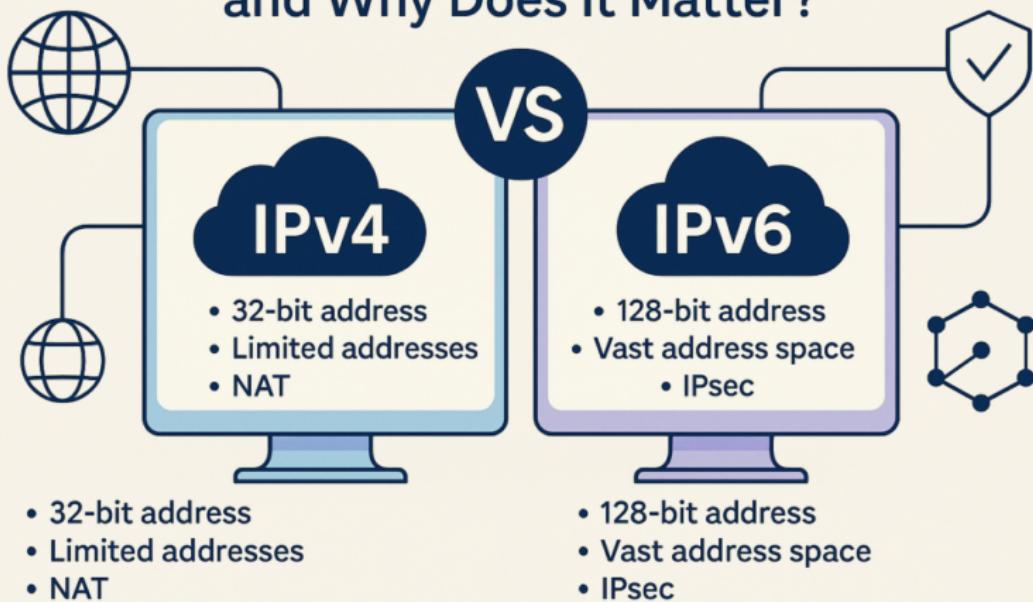
- Any other address (e.g., 8.8.8.8 or 142.250.184.142).
- These are globally unique and routable on the Internet.

# Understanding IPv6 Addresses

IPv4 ran out of addresses. IPv6 is the successor and has its own special types.

- **IPv6 (Internet Protocol v6):**
  - A **128-bit** address (e.g.,  
`2001:0db8:85a3 :: 8a2e:0370:7334`).
  - Provides a virtually limitless supply of addresses.
- **Special IPv6 Addresses:**
  - **Loopback:** `:: 1` (The equivalent of `127.0.0.1`).
  - **Link-Local:** `fe80 :: ...` (Automatically assigned for *local* LAN communication. Like ARP for IPv6).
  - **Unique Local:** `fd00 :: ...` (The equivalent of private IPv4 ranges).

## IPv4 vs IPv6: What's the Difference and Why Does It Matter?



## Addresses Aren't Enough: Ports

An IP address gets a data packet to the right *computer*. A **Port** gets it to the right *application* on that computer.

- **Analogy:** If an IP is the building's street address, the port is the apartment or office number.
- **Common Ports:**
  - 80: **HTTP** (Web)
  - 443: **HTTPS** (Secure Web)
  - 22: **SSH** (Secure Shell)

A connection is made to an **IP + Port** (e.g., 172.217.14.228:443).

## Leaving the LAN: The Default Gateway

1. Your computer (192.168.1.10) wants to send a packet to Google (8.8.8.8).
2. It looks at its subnet mask (255.255.255.0).
3. It realizes that 8.8.8.8 is **not** on its local network.
4. It can't send the packet directly. So, it sends it to the **Default Gateway**.

The **Default Gateway** is the IP address of the **Router** on the LAN (e.g., 192.168.1.1). It's the "door" out of your local network, responsible for forwarding all non-local traffic.

# Connecting Networks: Routing

Once the packet reaches your router, what's next?

- The **WAN (Wide Area Network)** is a network of networks (the Internet!).
- **Routing** is the process of finding the best path for data packets to travel from their source to their destination, hopping between thousands of different routers across the globe.

## The “Receptionist”: NAT i

**NAT (Network Address Translation)** is the clever workaround for the IPv4 address shortage.

- It allows an entire private network (e.g., all 50 devices in your home with 192.168.1.x addresses) to “hide” behind **one single public IP address**.
- Your router acts as a “receptionist,” keeping track of all outgoing requests and ensuring the responses get back to the correct private device.

Try using the [IPinfo](#) to identify your public IP.

## The “Phonebook”: DNS

We remember names (`google.com`), but computers only understand numbers (`142.250.184.142`).

**DNS (Domain Name System)** is the “phonebook of the Internet.” It is a global, distributed system that translates human-readable domain names into machine-readable IP addresses.

# More DNS Magic: mDNS & DDNS

- **mDNS (Multicast DNS):**
  - This is “local” DNS. It lets devices on your LAN find each other by name *without* a central DNS server.
  - This is how `my-laptop.local` or your printer automatically “appears.”
- **DDNS (Dynamic DNS):**
  - Your home’s public IP address can change (it’s “dynamic”).
  - DDNS is a service that automatically updates a domain name to point to your new IP address whenever it changes.
  - This is useful for hosting a server (e.g., Nextcloud) at home.

# Keeping the Network Running: NTP & SNMP

- **NTP (Network Time Protocol):**
  - Keeps the clocks on all computers and network devices in sync.
  - This is **critical** for security (encryption certificates), financial transactions, and accurate log files.
- **SNMP (Simple Network Management Protocol):**
  - Used by network administrators to monitor the health, performance, and configuration of routers, switches, and servers.

## What We Do on the Network i

Protocols are the “rules of conversation” for specific tasks.

- **HTTP (HyperText Transfer Protocol):** The fundamental protocol for the World Wide Web. It's how your browser *requests* web pages.
- **HTTPS (HTTP Secure):** This is just HTTP layered on top of **SSL/TLS** encryption. It ensures your communication is private and secure. **Always look for the 🔒 !**

# Email Protocols i

---

- **SMTP (Simple Mail Transfer Protocol):**
  - Used for *sending* email.
- **POP3 (Post Office Protocol):**
  - Used for *retrieving* email.
  - It *downloads* mail to your device and (usually) deletes it from the server. This is an older model.
- **IMAP (Internet Message Access Protocol):**
  - Used for *retrieving* email.
  - It *syncs* your mail with the server. This is the modern model. What you do on your phone appears on your laptop.

- **SSH (Secure Shell):**
  - The single most important tool for system administrators.
  - Provides an encrypted command-line connection to a remote server.
- **FTP (File Transfer Protocol):**
  - An old, *insecure* (plain text) protocol for transferring files.  
**Avoid it.**
  - Use **SFTP** (which runs over SSH) instead.
- **WebDAV / CalDAV / CardDAV:**
  - Extensions of HTTP that let you manage files (WebDAV), calendars (CalDAV), and contacts (CardDAV) on a web server. Used by services like Nextcloud.

- **MQTT (Message Queuing Telemetry Transport):**
  - A very lightweight and efficient protocol designed for “publish” and “subscribe” messages (a pub/sub model).
  - Perfect for **IoT (Internet of Things)**: sensors, smart bulbs, and small devices that need to send tiny messages reliably with low power.

# Network Configuration i

---

How does your device get an IP address?

- **Windows:**
  - Managed via the **Control Panel** or the Settings app.
- **Linux (Desktop):**
  - Almost always managed by **NetworkManager**, a user-friendly service with a graphical front-end (your network icon).
- **Linux (Server):**
  - Often managed by `systemd-networkd`.
  - Configuration is done via simple text files in `/etc/systemd/network/`.

## Automatic Configuration: DHCP i

In the last slide, we asked: "How does your device get an IP address?" For 99% of devices, the answer is **DHCP (Dynamic Host Configuration Protocol)**.

Manually setting an IP on every phone, laptop, and smart TV (a **static IP**) would be a nightmare. DHCP automates this.

1. Your device joins a network and shouts a **DHCP Discover** message: "Is there a DHCP server out there? I need an IP!"
2. A **DHCP Server** (usually your router) replies with a **DHCP Offer**: "Here, you can *use* 192.168.1.50."

## Automatic Configuration: DHCP ii

---

3. Your device accepts with a **DHCP Request**, and the server confirms with a **DHCP ACK** (Acknowledgement).

## DHCP Leases & IP Ranges

DHCP doesn't give you an IP forever. It gives you a **lease**.

- **Lease Time:** The IP is “rented” to your device for a specific time (e.g., 24 hours). Before it expires, your device must renew the lease. This ensures that IPs from devices that leave the network are eventually returned to the pool.
- **IP Range (Pool):** The DHCP server is configured to manage a *range* of addresses (e.g., 192.168.1.100 to 192.168.1.200).

## DHCP Leases & IP Ranges ii

---

- **Static Attribution:** By only using a range, the server leaves other IPs free (e.g., 192.168.1.1 to 192.168.1.99) for **static assignment**. These are manually configured on important devices like servers, printers, and the router itself, so their addresses never change.

## Diagnostic Tool 1: ping i

---

- **The Question:** "Are you there?"
- **The Action:** Sends a small packet (ICMP Echo Request) and waits for a reply.
- **The Answer:** Tells you if a host is reachable and how long the round-trip took (this is **latency**).
- **Example:** ping google.com

## Diagnostic Tool 2: traceroute i

---

- **The Question:** “What path do my packets take to get to you?”
- **The Action:** Sends packets with increasing “Time-To-Live” (TTL) values.
- **The Answer:** Shows you every single router (or “hop”) your packet passes through on its way to the destination. Great for finding *where* a connection is failing.
- **Example:** traceroute google.com

## Diagnostic Tool 3: ip, dig, nmap i

---

- `ip addr show`
  - The modern Linux tool to view your *own* IP configuration and network interfaces. (Replaces the old `ifconfig`).
- `dig google.com`
  - **The Question:** "What is the IP address for this name?"
  - **The Action:** Performs a DNS lookup.
- `nmap localhost`
  - **The Question:** "What doors are open on this machine?"
  - **The Action:** A powerful port scanner.
  - **The Answer:** Reports which ports are open and what services are (likely) running on them.

# Network Monitoring: Wireshark i

---

- **The Tool:** Wireshark is a “network sniffer” or protocol analyzer.
- **The Action:** It captures *every single packet* traveling on your network interface and lets you inspect its contents.
- **Analogy:** It’s like a video camera for your network traffic.
- **Use:** The single most powerful tool for debugging complex network problems.

## Network Security: Firewalls

A **firewall** is the “security guard” for your network or computer.

- It inspects all incoming and outgoing network traffic.
- It decides whether to **allow** or **block** each packet based on a set of rules (e.g., “Allow traffic on port 443, block everything else”).
- This is your first line of defense.

## Firewall Examples i

---

- **iptables:** The classic, powerful, command-line firewall built into the Linux kernel for decades.
- **nftables:** The modern successor to iptables in Linux. It has a simpler syntax and better performance.
- **pfSense:** A free, open-source **firewall operating system**. You install it on a dedicated computer to turn it into an extremely powerful, enterprise-grade router and firewall for your entire network.

# The Power of SSH: VS Code Remote

---

SSH is more than just a remote shell. The “Remote - SSH” extension in VS Code is a game-changer.

- **How it works:** Your editor UI runs locally, but all file editing, terminal commands, and language processing run on the remote server.
- You get the power of a server with the comfort of your local editor.

# The Power of SSH: Tunnels & X11

---

- **SSH Tunnels (Port Forwarding):**
  - Lets you securely “wrap” network traffic inside an SSH connection.
  - **Example:** Access a database running on `localhost:5432` on a remote server as if it were running on *your own* machine’s `localhost:5432`.
- **X11 Forwarding:**
  - Lets you run a *graphical* application (like `firefox` or a text editor) on a remote Linux server, but see and interact with the window on your local desktop.

# Syncing Files: rsync over SSH in

---

The best way to transfer and sync files. `rsync` is fast, efficient, and versatile.

- **Why it's fast:** It only copies the **differences** (deltas) between files, not the whole file.
- It works perfectly over an SSH connection.
- **Command:**

```
# Sync a local folder UP to a remote server
$ rsync -avzP ./my-project/ user@host:~/projects/
```

# The Modern Web: Reverse Proxy i

---

A **Reverse Proxy** (like NGINX or Caddy) is a server that sits *in front* of your actual application servers.

- It receives all incoming traffic from the Internet.
- It then “proxies” (forwards) the request to the correct internal application (e.g., your Python app, your Node.js app).
- **Uses:**
  - **Load Balancing:** Distribute traffic across multiple app servers.
  - **Security:** Hides your application servers from the Internet.
  - **Hosting:** Host multiple websites on a single IP address.

# The Modern Web: HTTPS & Let's Encrypt i

---

- **HTTPS** is essential. It provides the encryption (SSL/TLS) that keeps user data private and proves your site's identity.
- **The Problem:** Certificates used to be expensive and difficult to install.
- **The Solution: Let's Encrypt**
  - A free, automated, and open **Certificate Authority (CA)**.
  - It provides free SSL/TLS certificates and tools (like certbot) to automatically install and renew them.
  - It has made the entire web more secure.

## Further Reading & Resources i

- **Wireshark:** <https://www.wireshark.org/>
- **Nmap:** <https://nmap.org/>
- **Let's Encrypt:** <https://letsencrypt.org/>
- **ip command guide:** <https://www.geeksforgeeks.org/ip-command-in-linux-with-examples/>
- **rsync guide:**  
<https://www.digitalocean.com/community/tutorials/how-to-use-rsync-to-sync-local-and-remote-directories>
- **Mozilla's Guide to HTTP:**  
<https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>