

# Homework — What the Internet Actually Is

Estimated time: ~45 min · No code required

Three short tool-tour exercises. Goal: turn the abstract picture from the theory page (DNS → TCP+TLS → HTTP → CDN) into something you've actually *seen* on your own machine. You'll inspect your DNS resolver, watch a real page load request-by-request in your browser, and read a TLS certificate. Take a screenshot at each step.

## How to do these

Open each linked site in a new tab. No installs and no accounts required. Tasks 2 and 3 are easiest in **Chrome**, **Edge**, or **Firefox** on a desktop / laptop — mobile browsers don't expose dev-tools the same way. If a step doesn't match what you see exactly, write down what's *different* — browsers update their UIs constantly.

## Task 1 — See your own DNS

First we'll find out which DNS resolver your computer is currently using, and what the public internet sees as your IP address.

1. Open [dnsleaktest.com](https://dnsleaktest.com). Click the **"Standard test"** button. After a few seconds you'll see a list of one or more DNS server IPs and the owning *ISP / provider* (e.g., [Google](#), [Cloudflare](#), your home ISP, your VPN provider).
2. Write down: the DNS server IP(s) and the owning provider name.
3. Now open [ipinfo.io](https://ipinfo.io). Write down: your **IP address**, your **ISP** (the "org" line), and your **approximate city**.
4. Bonus (optional): if a friend or family member is in a different country, ask them to do the same two steps and compare. Different country → different ISP → almost certainly a different resolver.

✓ **Self-check:** you can answer *"what DNS resolver am I using right now?"* with a specific name (e.g., *"Cloudflare 1.1.1.1"* or *"my home ISP's resolver"*); you

can answer *"is my IP a residential or VPN IP?"* based on the ISP / org line; you understand that **both** answers can change depending on the network you're on (home vs. cafe vs. office vs. VPN).

## Task 2 — Inspect a real request

Now we'll watch a real page load, request-by-request, using the browser's built-in dev-tools.

1. Open [wikipedia.org](https://wikipedia.org) in Chrome, Edge, or Firefox.
2. Right-click anywhere on the page → **Inspect** (or press `F12` / `Cmd-Opt-I` on macOS, `Ctrl-Shift-I` elsewhere).
3. Switch to the **Network** tab in the panel that appears.
4. **Reload the page** (`Cmd-R` on macOS, `Ctrl-R` elsewhere). You'll see a list of every individual request the browser made — HTML, CSS, JS, images, fonts.
5. Sort the list by **Size** (click the column header). Find the **largest single request**. Click it.
6. In the side panel that opens, look at the **Headers** tab. Identify:
  - the HTTP **method** (GET, POST, ...);
  - the response **status code** (200 OK, 304 Not Modified, ...);
  - the `Content-Type` response header (e.g., `image/jpeg`, `text/html`, `application/javascript`);
  - approximately **how many total requests** the page made (look at the bottom-left summary line — *"N requests"*).
7. Take a screenshot of the Network tab.

✓ **Self-check:** you can name (a) the HTTP method, (b) the status code, (c) the `Content-Type` of the largest request, and (d) the rough total number of requests a real page makes (usually dozens to hundreds — not one). You now know where to look the next time someone says *"the page is slow"* — the **Network** tab is the first place to check.

### Task 3 — Trace the certificate

Finally, we'll read the TLS certificate that makes the padlock icon appear in your URL bar.

1. Still on [wikipedia.org](https://wikipedia.org), click the **padlock icon** in the URL bar.
2. Click "Connection is secure" → "Certificate is valid" (Chrome / Edge wording) or "More information" → "View Certificate" (Firefox). A certificate viewer opens.
3. Identify and write down:
  - the **Issuer** — this is the Certificate Authority (CA) that signed it (e.g., "Let's Encrypt", "DigiCert", "GlobalSign");
  - the **"Valid until" / "Not After"** date — this is when the certificate expires;
  - the **Subject** or **Common Name** — the hostname(s) this certificate is valid for (often a wildcard like `*.wikipedia.org` or a SAN list).
4. Repeat on a smaller site — your bank, your school, a local shop's site. Notice the differences: different CAs are common; some banks still use **EV (Extended Validation)** certificates where the legal entity name appears in the certificate detail; expiry windows vary (Let's Encrypt = 90 days, others = 1 year).

✓ **Self-check:** for both sites you inspected, you can answer (a) *who issued the cert*, (b) *when it expires*, (c) *for which hostname(s) it's valid*. You understand that the padlock means *"the connection is encrypted AND the cert's name matches the URL bar"* — **not** *"this site is trustworthy."*

## What "done" looks like

You finish this homework when you have, in your notes:

- The DNS resolver IP + provider you're currently using, plus your public IP and ISP from `ipinfo.io` (Task 1).
- A screenshot of the Network tab on a real Wikipedia load, plus the method / status / Content-Type / total-request count of the largest single request (Task 2).

- The CA, expiry date, and hostname(s) for the Wikipedia certificate, plus the same three for one other site (Task 3).

That's the whole DNS → TCP+TLS → HTTP chain from the theory page, observed once on your own machine. The next time a site is slow, broken, or warns you about a certificate, you'll know which of these three layers to look at first.