Chapter 1.4: The Internet

Computer Fundamentals and Introduction to Programming

Network basics | Addressing | The Web | Digital services | Security

Sercan KÜLCÜ · Giresun University

© Sercan Külcü. All rights reserved. · Interactive HTML version of these slides: https://sercankulcu.github.io/files/computer/slides/Chapter_01_04_The_Internet.html

Learning Outcomes

Network concepts

Explain how the Internet, clients, servers and protocols relate.

Addressing

Distinguish IP addresses, domain names, DNS and URLs.

Using the Web

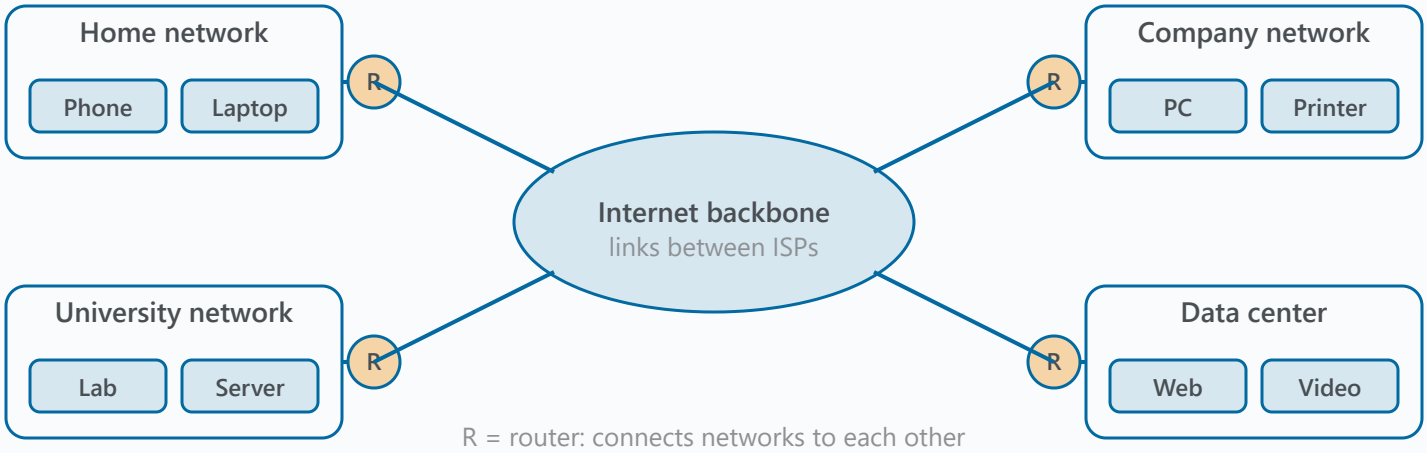
Use browsers and search engines effectively.

Security

Recognize threats and build safe habits.

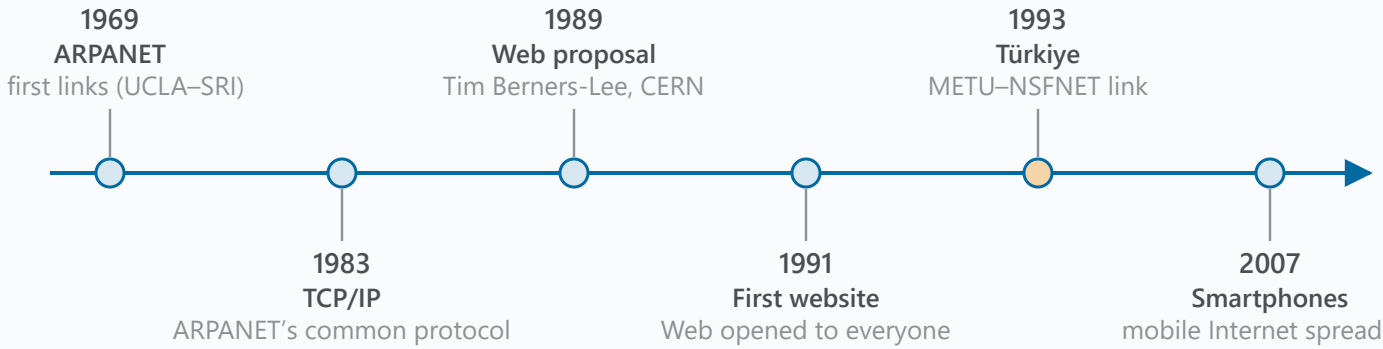
What Is the Internet?

A computer network connects devices so they can share data and resources. The Internet is a global **network of networks** that joins networks of every size under common rules (TCP/IP).



NETWORK BASICS

How the Internet Grew

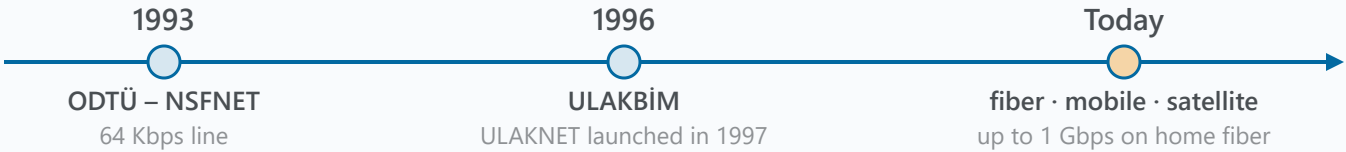


The Web and the Internet are not the same: the Internet is an infrastructure dating from 1969; the Web is a service that started on top of it in 1991. Today mobile networks, fiber and cloud services keep extending global access.

NETWORK BASICS

The Internet in Türkiye

Türkiye's first permanent Internet connection was set up between METU (ODTÜ) and NSFNET on April 12, 1993.



1993

Academic access began over a 64 Kbps link.

1996

TÜBİTAK ULAKBİM was founded; the ULAKNET academic network went live in 1997.

Today

Fiber, mobile and satellite links offer different ways to connect.

Getting Online

Fiber

High capacity and low latency.

DSL

A fixed connection over the phone line.

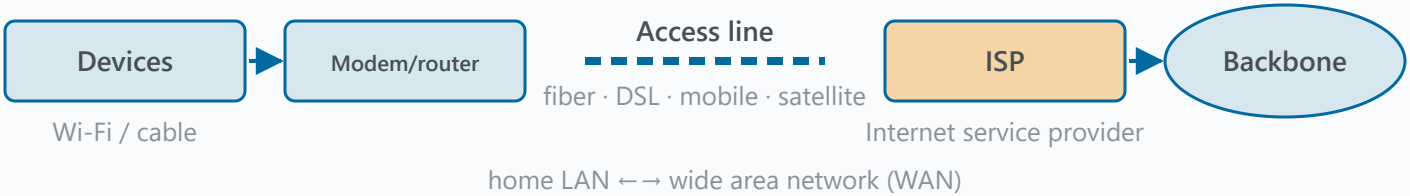
Mobile data

Portable access through cell towers.

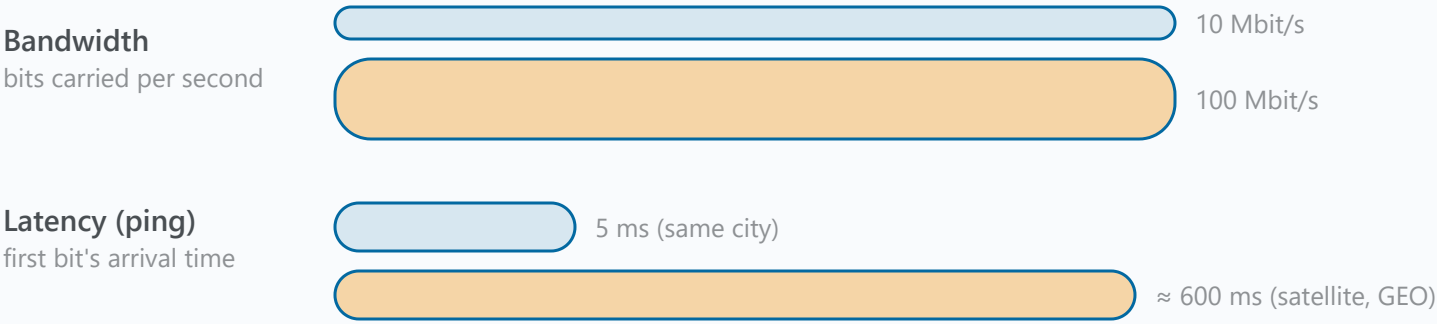
Fixed wireless and satellite

An option where cables are hard to run.

An **ISP** (Internet service provider) connects users to the Internet backbone.



Bandwidth and Latency



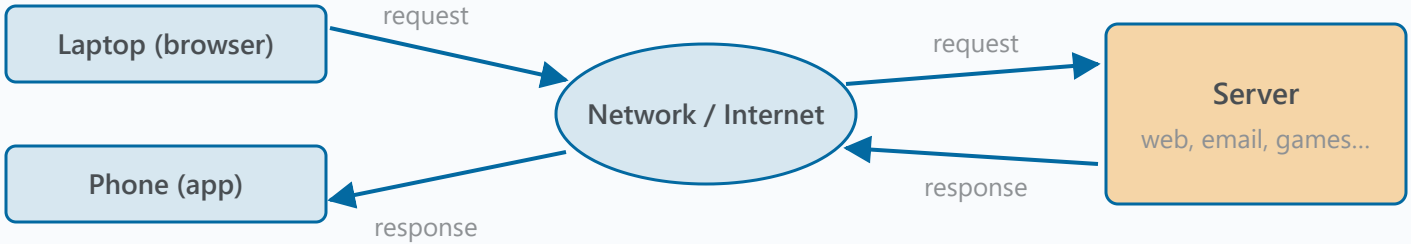
Download time

On a 100 Mbit/s line, a 500 MB file downloads in at best 500 ÷ 12.5 MB/s = **40 s**.

Which matters?

Bandwidth for large downloads; low latency for gaming and video calls.

Clients, Servers and Protocols



Client

A device or program that requests a service.

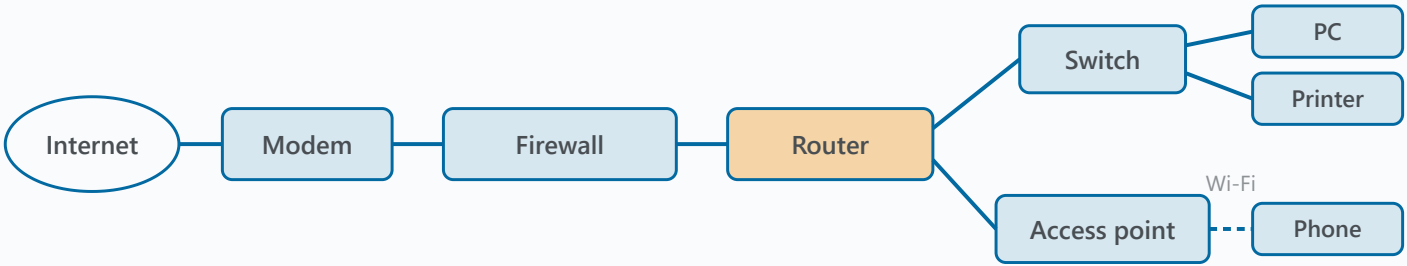
Server

A system that answers requests and provides resources.

Protocol

Rules for the format and order of communication.

Network Devices



At home, the modem, router, switch and access point are often one box; every device has a NIC.

NIC

Connects a device to the network.

Switch

Forwards frames to the right device on the local network.

Router

Routes packets between different networks.

Modem

Converts the signal on the access line into network data.

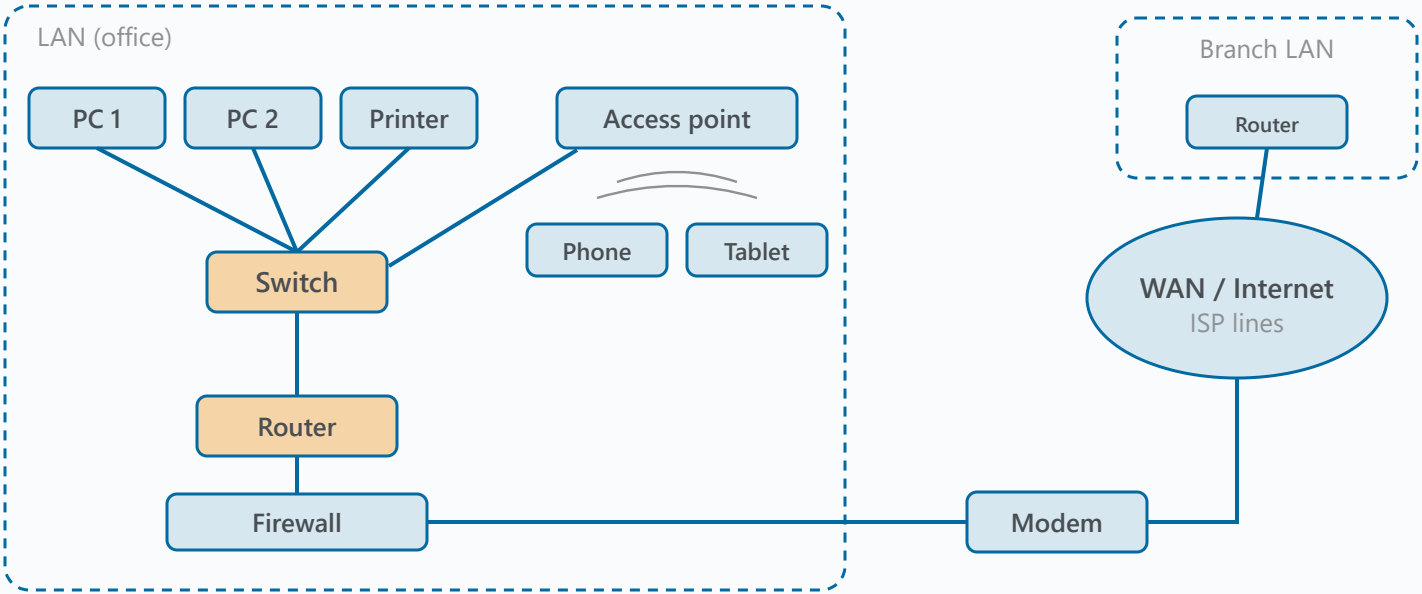
Access point

Connects wireless devices to the local network.

Firewall

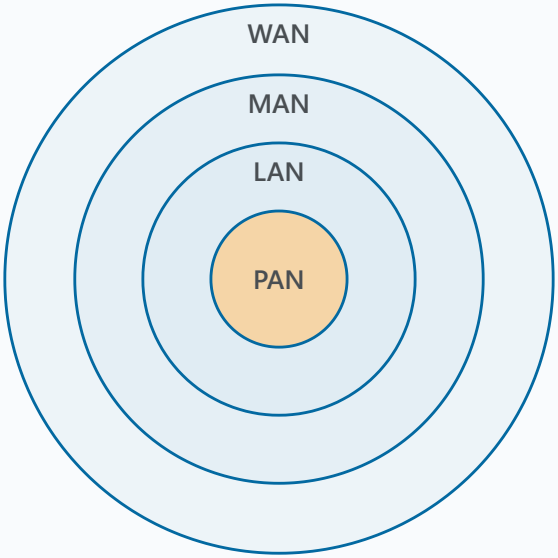
Filters network traffic according to rules.

A Typical Network: From LAN to WAN



A switch connects devices on the same local network; a router connects different networks. At home, the modem, router, switch and access point usually share one box.

Geographic Scope of Networks



Type	Scope	Example
PAN	Personal area	Phone and earbuds
LAN	Room, building or campus	Office network
MAN	City scale	Municipal network
WAN	Country or continent	A company's branches; the Internet is the largest WAN

NETWORK BASICS

Wi-Fi Standards

Name	IEEE	Bands	Main difference
Wi-Fi 5	802.11ac	5 GHz	Wide channels; much faster than 802.11n
Wi-Fi 6	802.11ax	2.4 and 5 GHz	More efficient in crowded places (OFDMA); better with many devices
Wi-Fi 6E	802.11ax	2.4, 5 and 6 GHz	Wi-Fi 6 plus the 6 GHz band; less interference
Wi-Fi 7	802.11be	2.4, 5 and 6 GHz	Wider channels (320 MHz), simultaneous links on several bands (MLO)
4G (LTE) → 5G	3GPP	Carrier bands	5G typically brings higher speed and lower latency; real values depend on the carrier and coverage

Theoretical speeds rise with each generation; real speed depends on distance, obstacles, channel width and support on both the router and the device.

NETWORK BASICS

Wi-Fi Bands

2.4 GHz

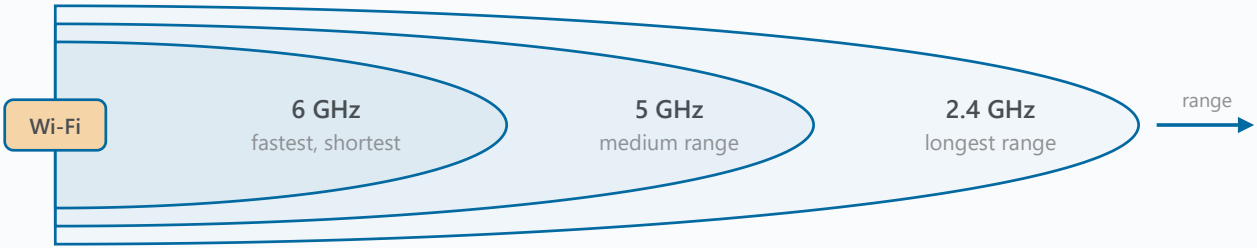
Long range, passes walls better; usually slower. Crowded by microwave ovens, Bluetooth and neighboring networks.

5 GHz

Medium range, higher speed. Weakens more through walls than 2.4 GHz.

6 GHz

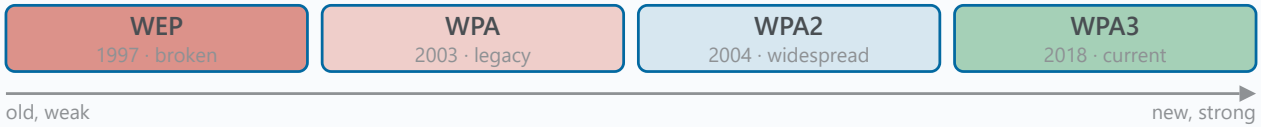
Short range, highest potential speed, little congestion (Wi-Fi 6E and 7). Most affected by obstacles.



The general trade-off: **higher frequency means more potential speed but less range and wall penetration.** In a far room 2.4 GHz often works better; up close, 5/6 GHz.

Wi-Fi Security

Protocol	Status	Note
WEP	Insecure	Very old; can be cracked in minutes. Should no longer be used.
WPA / TKIP	Legacy	Not recommended; exists only for old devices.
WPA2 (AES)	Widespread	Needs a strong, long password; weak ones can be guessed.
WPA3	Current	More resistant to weak passwords; the 6 GHz band uses WPA3 (or Enhanced Open). Requires device support.



Open (password-free) networks: traffic may be unencrypted; someone on the same network can eavesdrop or set up a fake access point. For sensitive tasks (banking, government services) use HTTPS and, if needed, a VPN.

Communication Protocols

IP

Addressing and routing packets between networks.

TCP

Ordered, reliable delivery of data.

UDP

Low-latency, connectionless delivery.

DNS

Translating domain names into IP addresses.

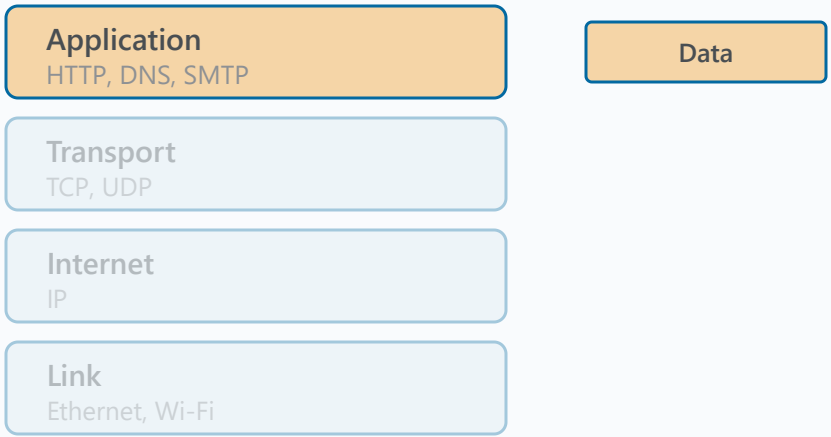
HTTP/HTTPS

Transferring web resources. HTTPS encrypts the communication.

SMTP/IMAP

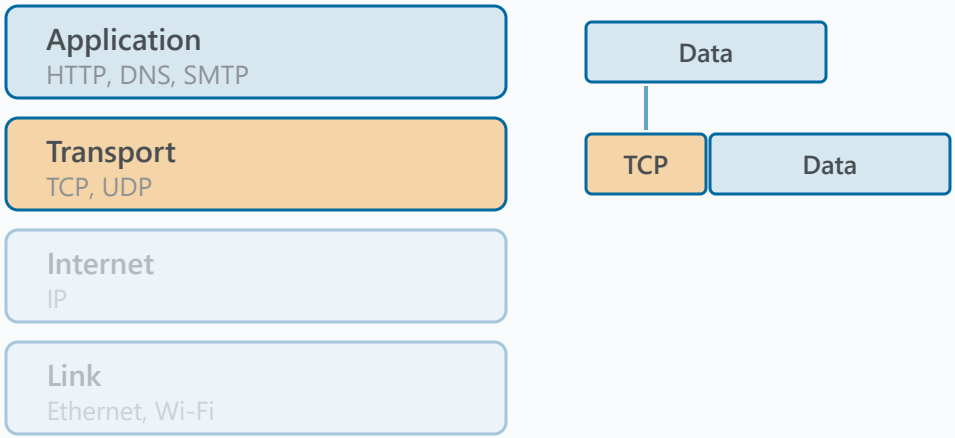
Sending and receiving email.

Protocols Work in Layers



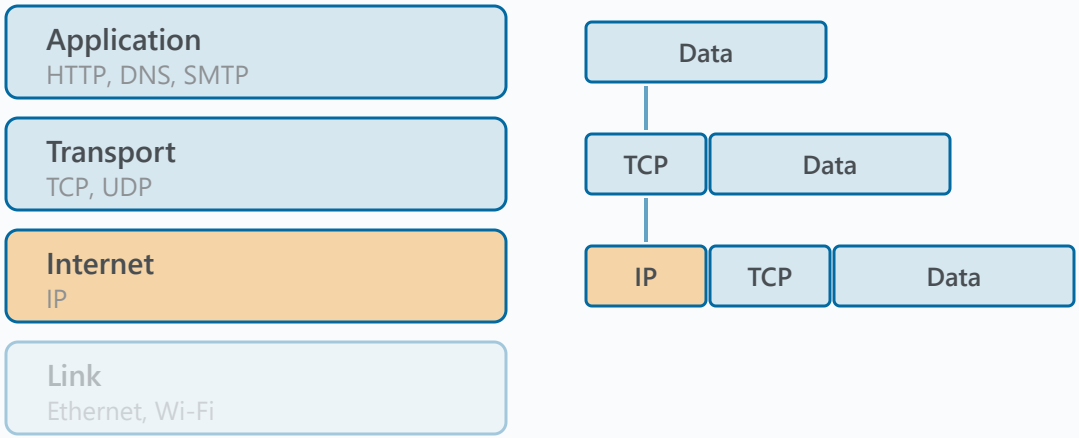
The browser produces an HTTP request (data). The application layer decides “what to say”.

Protocols Work in Layers



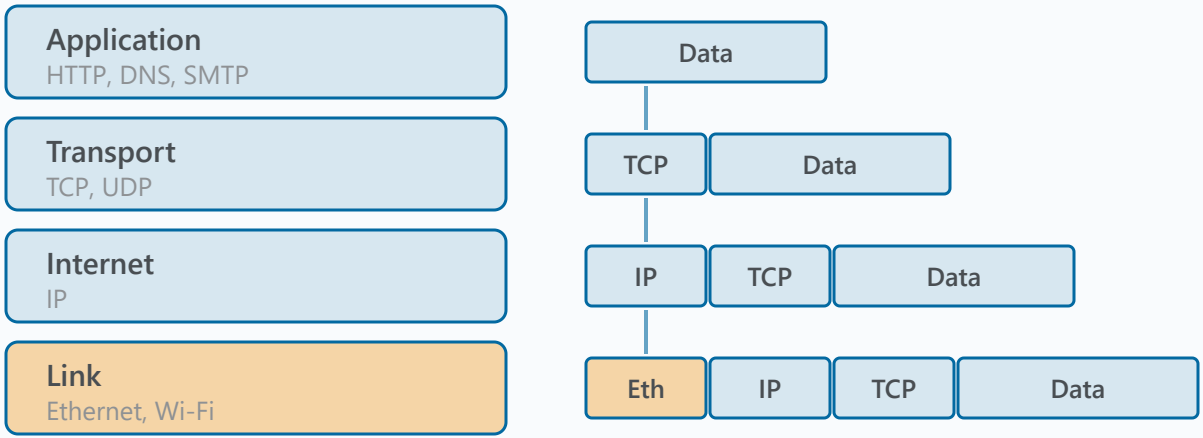
The transport layer splits the data and adds a **TCP header** to each piece: port number, sequence number. A lost piece can be requested again.

Protocols Work in Layers



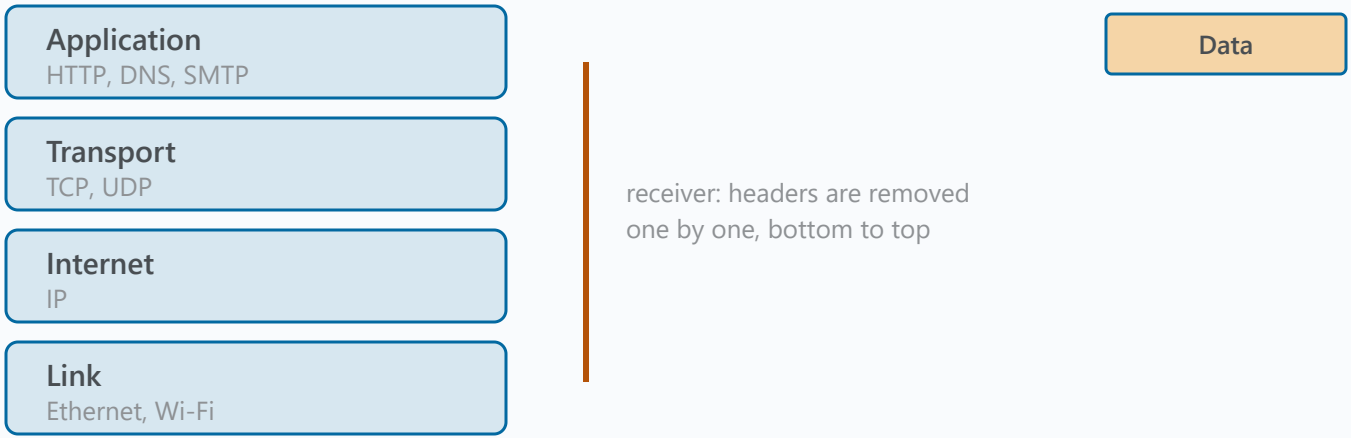
The Internet layer adds an **IP header**: source and destination IP addresses. Routers choose a path by looking only at this header.

Protocols Work in Layers



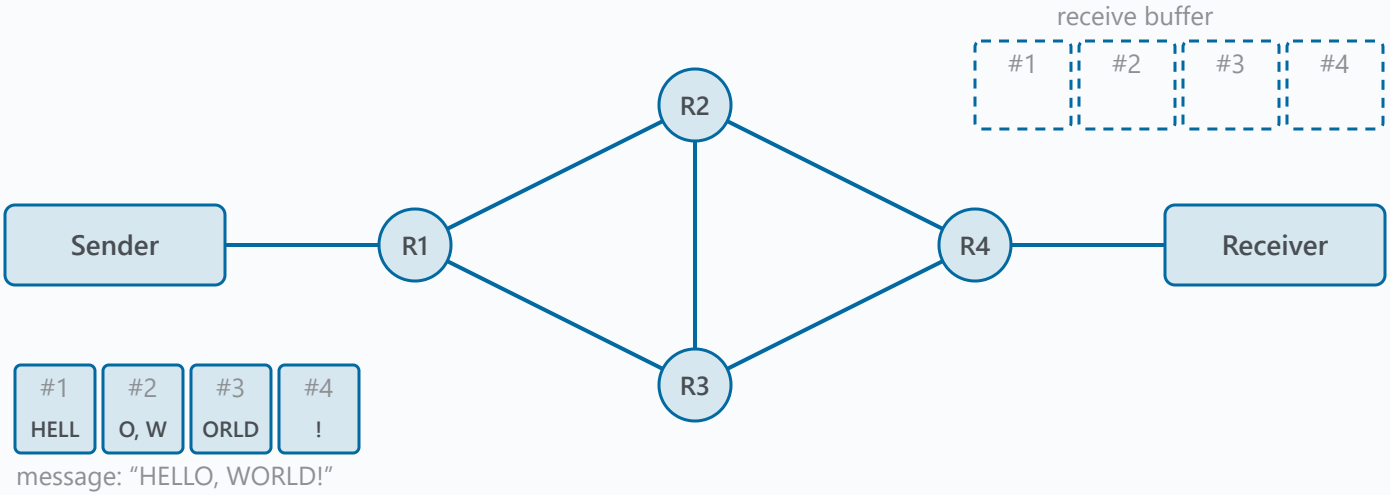
The link layer adds a **frame** header (MAC address) and sends the bits over a cable or radio.

Protocols Work in Layers



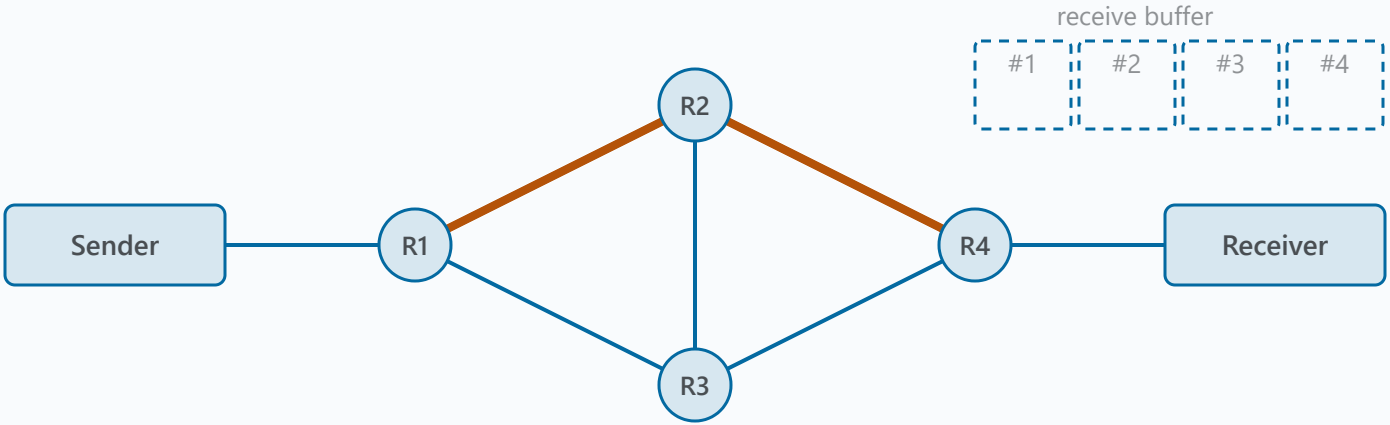
At the receiver the process reverses: each layer reads and strips its own header, and the data reaches the server application just as the browser sent it.

Data Travels in Packets



The message is not sent in one piece: it is split into **4 packets**. Each packet's header holds a sequence number and the source and destination addresses.

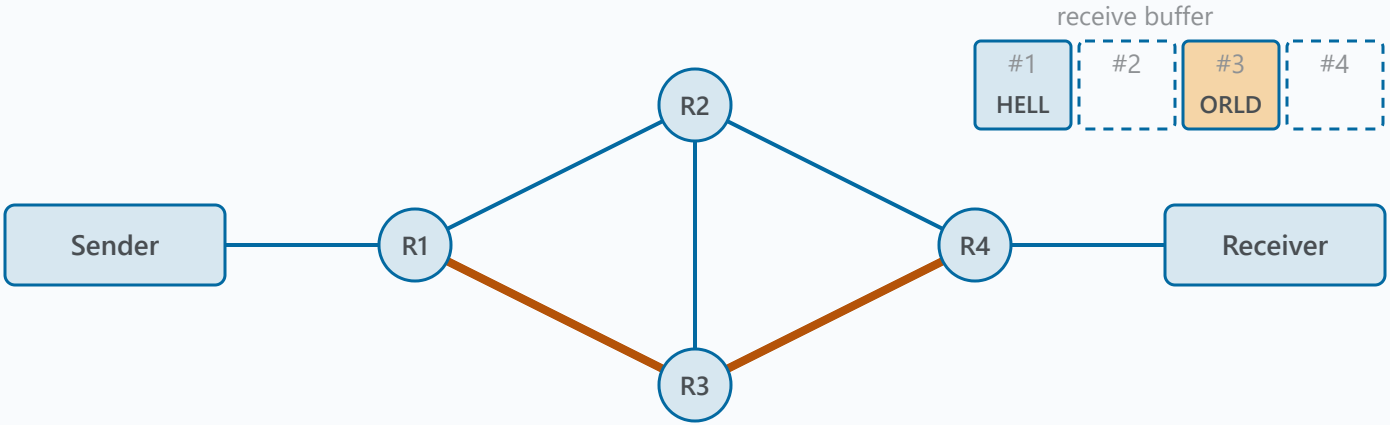
Data Travels in Packets



message: “HELLO, WORLD!”

Packets travel independently: #1 and #3 take R1 → R2 → R4.

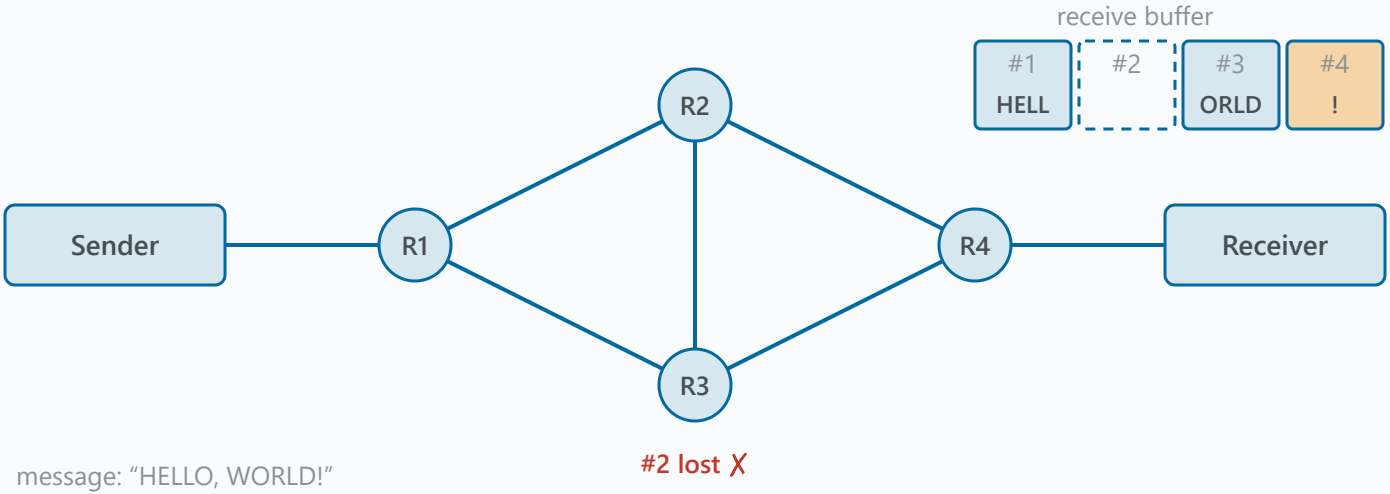
Data Travels in Packets



message: “HELLO, WORLD!”

#2 and #4 take R1 → R3 → R4. Routers choose the best path for each packet at that moment.

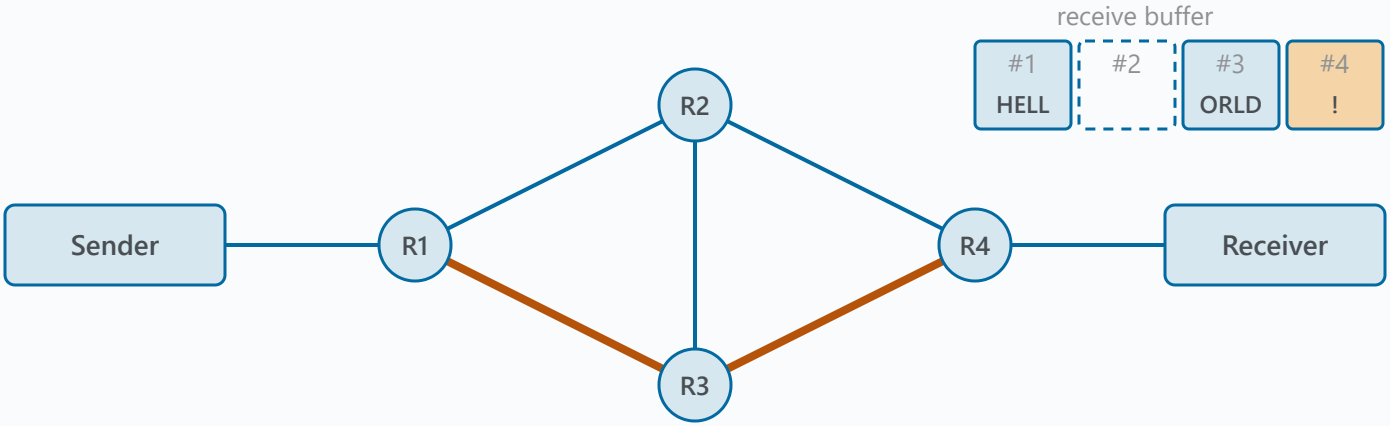
Data Travels in Packets



message: “HELLO, WORLD!”

Packets arrive out of order: first #1, #3, #4. #2 was lost at a busy router.

Data Travels in Packets

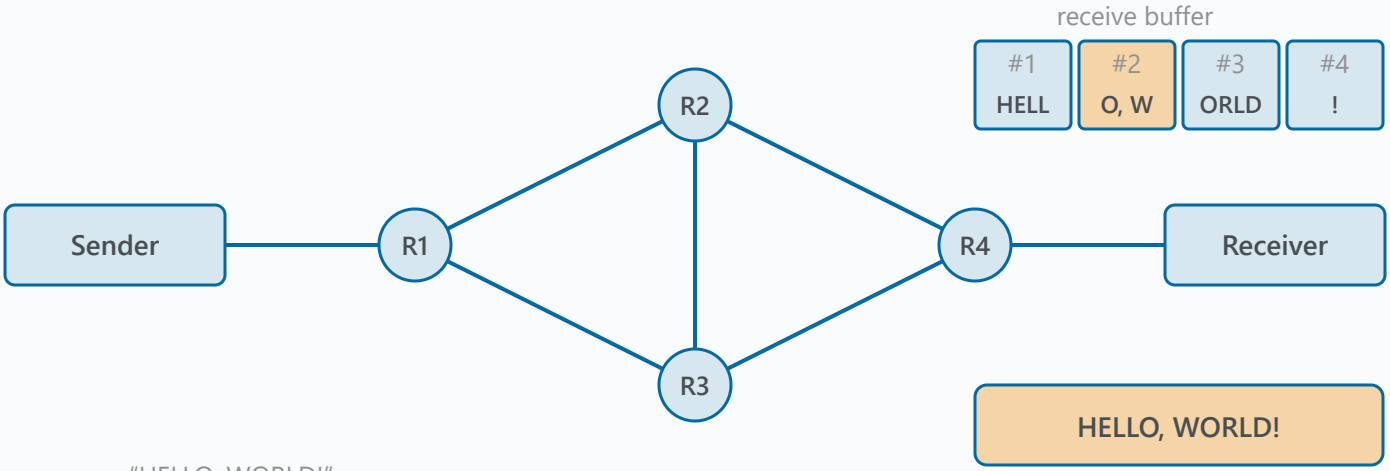


message: “HELLO, WORLD!”

receiver: “#2 missing, resend”

TCP notices the missing sequence number; #2 is resent (via R3 again).

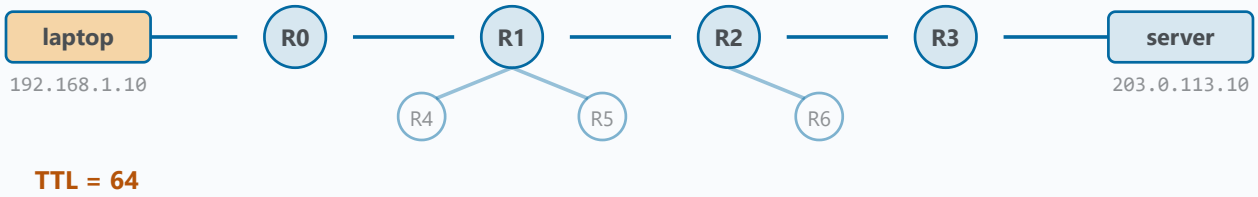
Data Travels in Packets



message: “HELLO, WORLD!”

All packets have arrived. The receiver orders them by sequence number and reassembles the message: “HELLO, WORLD!”.

A Packet's Path Through Routers, Step by Step



The laptop (192.168.1.10) sends a packet to the server at **203.0.113.10**. The IP header has **TTL = 64**. The destination is not on the local network → the packet goes to the default gateway (R0, the home router).

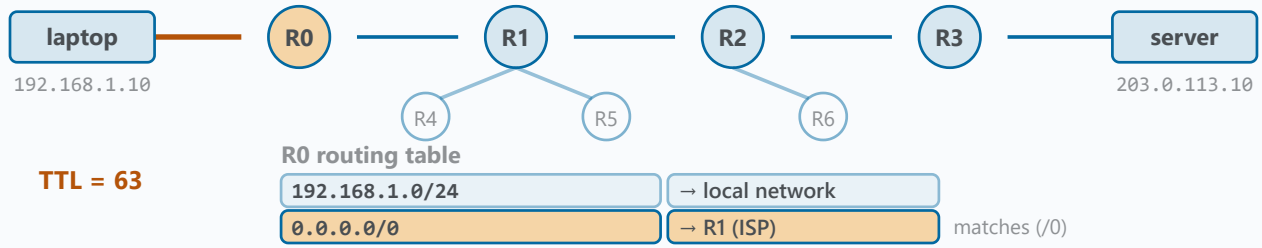
Step 1 / 6

```
1 receive packet, look at destination IP
2 TTL = TTL - 1;  if 0, drop and notify source
3 find table entries that match the destination
4 pick the longest-prefix (most specific) entry
5 forward to that entry's next hop
```

VARIABLES

at = **laptop** TTL = **64** next = **R0**

A Packet's Path Through Routers, Step by Step



R0 receives the packet and decrements TTL (63). The destination matches only **0.0.0.0/0** (the default route) → the ISP's router R1. (With NAT, the source address is also changed here.)

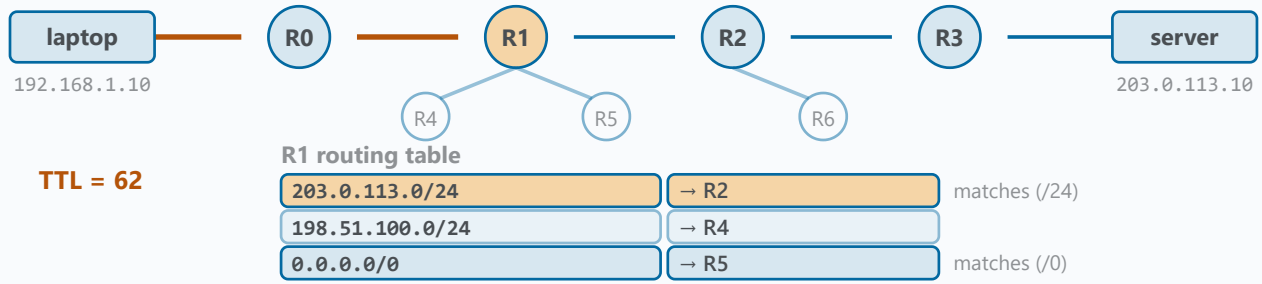
Step 2 / 6

```
1 receive packet, look at destination IP
▶ TTL = TTL - 1;  if 0, drop and notify source
▶ find table entries that match the destination
▶ pick the longest-prefix (most specific) entry
5 forward to that entry's next hop
```

VARIABLES

`at = R0` `TTL = 63` `match = 0.0.0.0/0` `next = R1 (ISP)`

A Packet's Path Through Routers, Step by Step



R1: TTL 62. Two entries match: /24 and /0. **Longest prefix wins**: 203.0.113.0/24 is more specific → R2. The routes to R4 and R5 are not used.

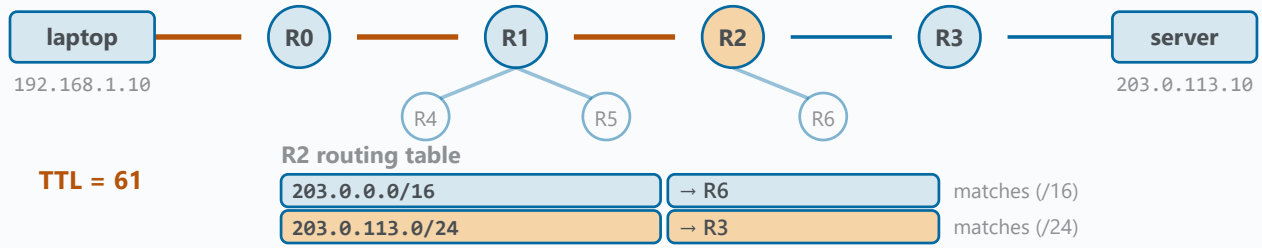
Step 3 / 6

```
1 receive packet, look at destination IP
▶ TTL = TTL - 1;  if 0, drop and notify source
▶ find table entries that match the destination
▶ pick the longest-prefix (most specific) entry
5 forward to that entry's next hop
```

VARIABLES

at = R1 · TTL = 62 · match = 203.0.113.0/24 · next = R2

A Packet's Path Through Routers, Step by Step



R2: TTL 61. Both 203.0.0.0/16 and 203.0.113.0/24 match; /24 is longer → **R3**. Had /16 been chosen, the packet would have gone to R6.

Step 4 / 6

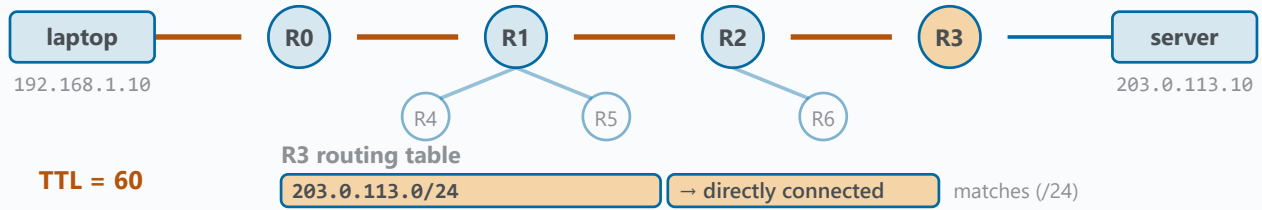
```
1 receive packet, look at destination IP
▶ TTL = TTL - 1;  if 0, drop and notify source
▶ find table entries that match the destination
▶ pick the longest-prefix (most specific) entry
5 forward to that entry's next hop
```

VARIABLES

at = **R2** TTL = **61**

match = **203.0.113.0/24** next = **R3**

A Packet's Path Through Routers, Step by Step



R3: TTL 60. The destination network is **directly connected** to R3; the packet is delivered to the server.

Step 5 / 6

```
1 receive packet, look at destination IP
▶ TTL = TTL - 1;  if 0, drop and notify source
▶ find table entries that match the destination
4 pick the longest-prefix (most specific) entry
5 forward to that entry's next hop
```

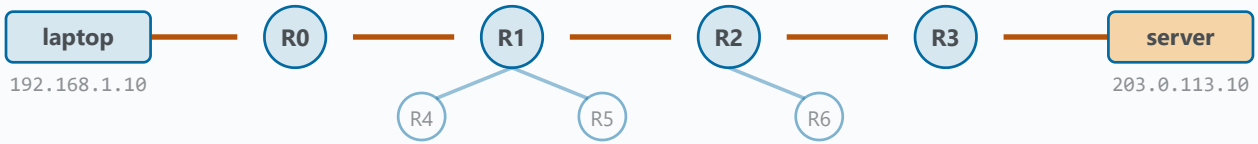
VARIABLES

at = **R3** TTL = **60**

match = **203.0.113.0/24**

next = **directly connected**

A Packet's Path Through Routers, Step by Step



The packet crossed 4 routers and TTL dropped from 64 to 60. TTL keeps a packet caught in a loop from circulating forever: when it hits 0, the packet is dropped and a “time exceeded” message goes back to the sender. **tracert** / **traceroute** sends packets with TTL 1, 2, 3… to list the routers on the path.

Step 6 / 6

```
1 receive packet, look at destination IP
2 TTL = TTL - 1;  if 0, drop and notify source
3 find table entries that match the destination
4 pick the longest-prefix (most specific) entry
▶ forward to that entry's next hop
```

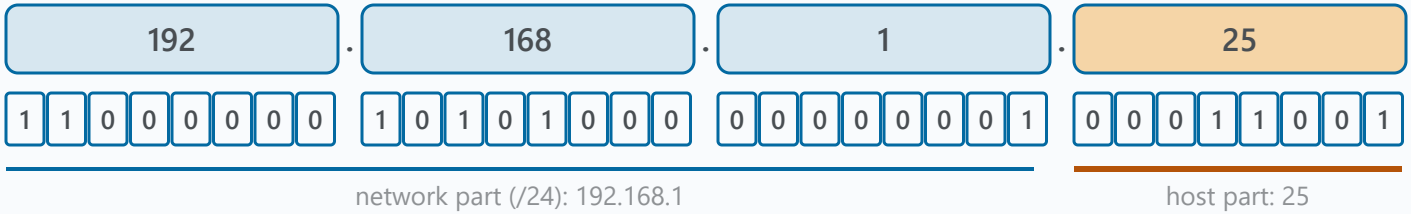
VARIABLES

at = **server** | TTL = **60** | next = **–**

ADDRESSING

IP Addresses

An IP address identifies a network interface and lets packets be routed to it.



IPv4

32 bits: four numbers separated by dots (each 0–255), called dotted-decimal notation.

`192.168.1.25`

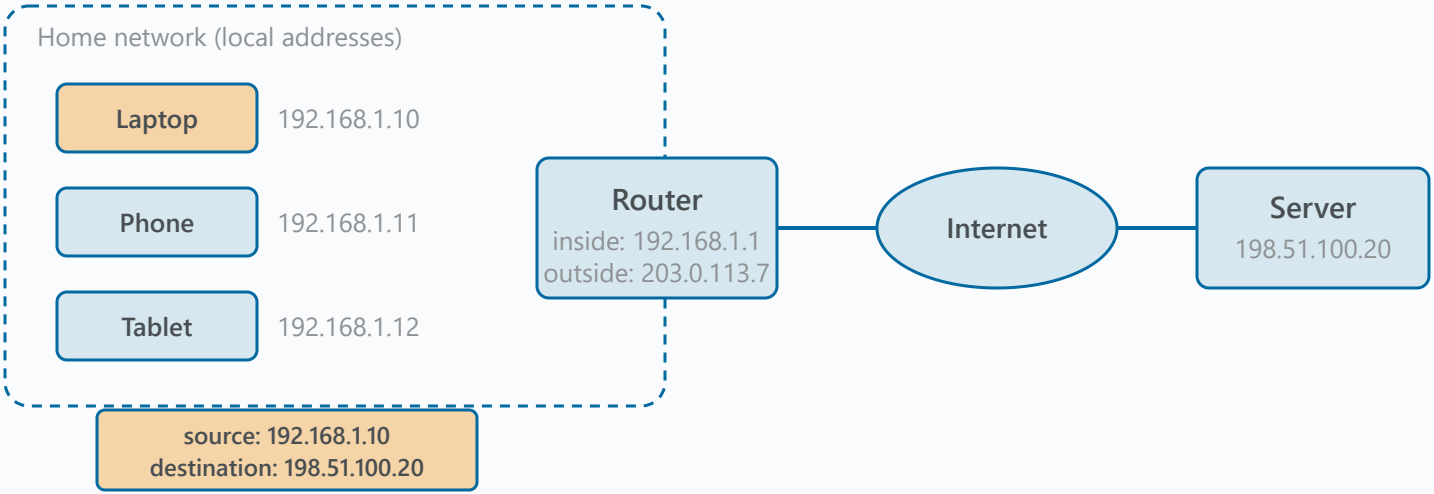
IPv6

128 bits: groups of hex digits; a much larger address space.

`2001:db8::25`

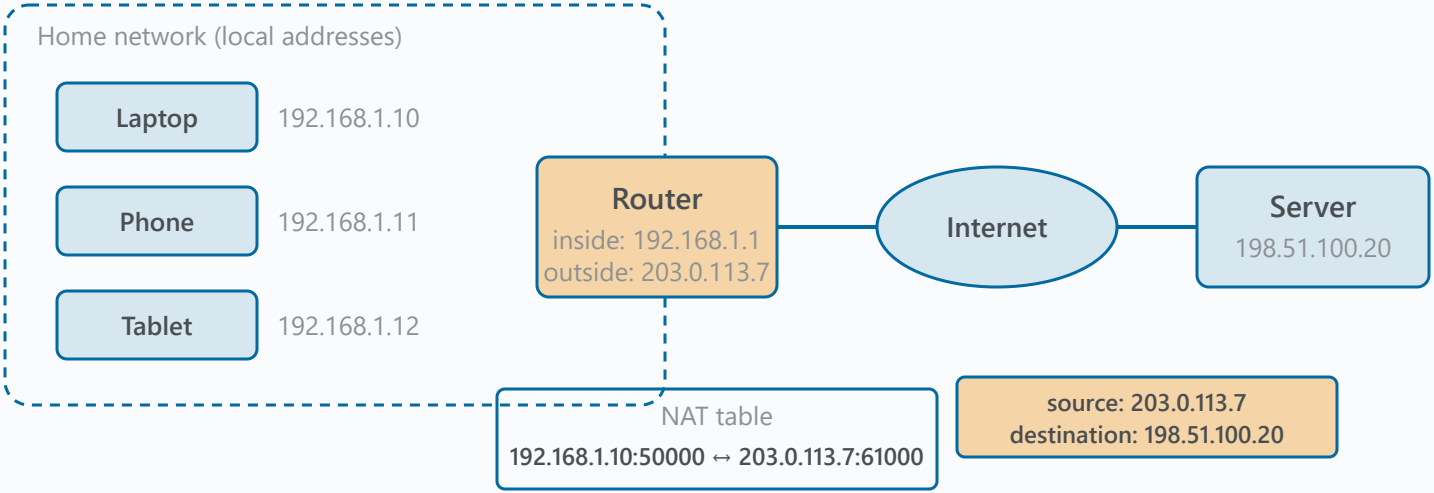
A device's local IP address need not be the same as the public IP address seen on the Internet.

Local and Public IP Addresses (NAT)



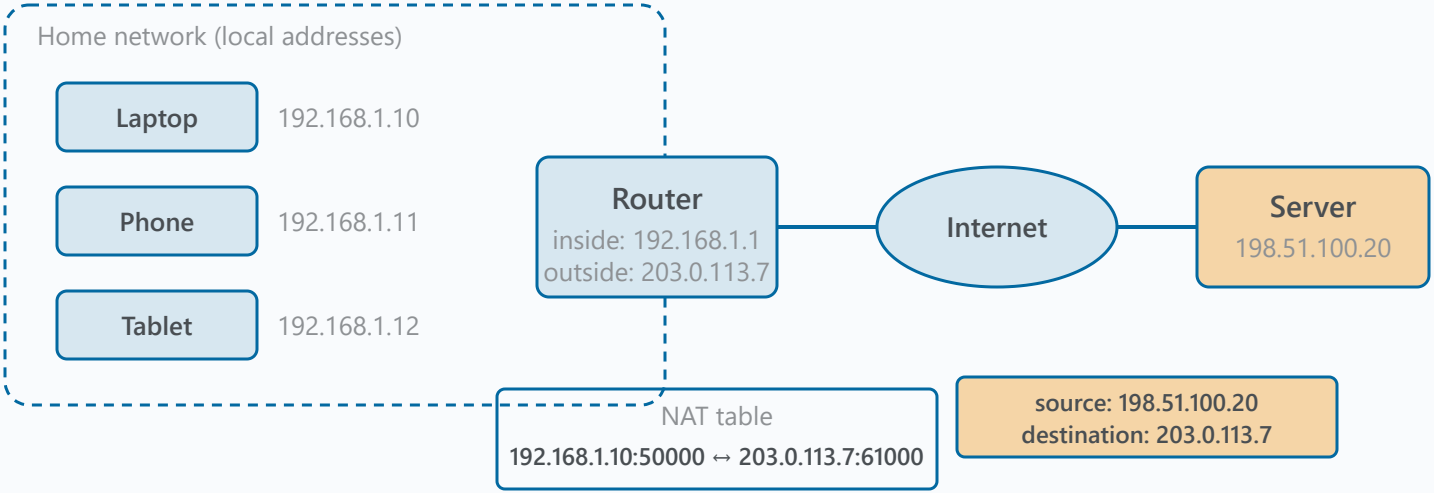
The laptop sends a packet to the server. The source is its **local** IP (192.168.1.10); private addresses are not routed on the Internet. The router replaces it (together with the port number, NAPT) with a public IP.

Local and Public IP Addresses (NAT)



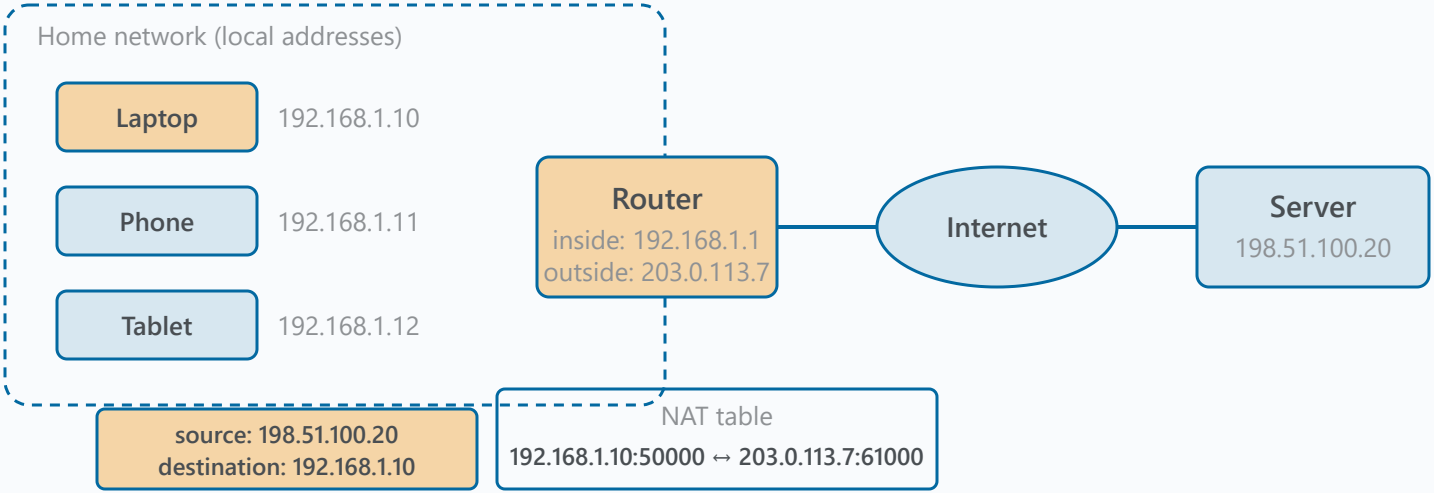
The router replaces the source address with its own **public** IP (203.0.113.7) and records the mapping in its NAT table.

Local and Public IP Addresses (NAT)



The server sees only 203.0.113.7 and sends its reply there. From outside, every device in the home looks like one address.

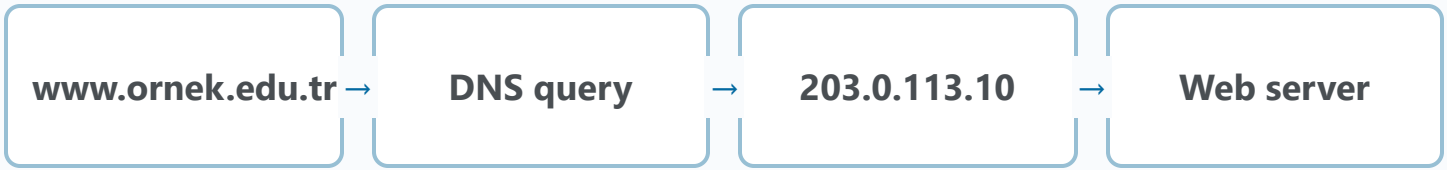
Local and Public IP Addresses (NAT)



The router checks its NAT table: the reply belongs to the laptop at 192.168.1.10. It rewrites the destination and forwards the packet.

ADDRESSING

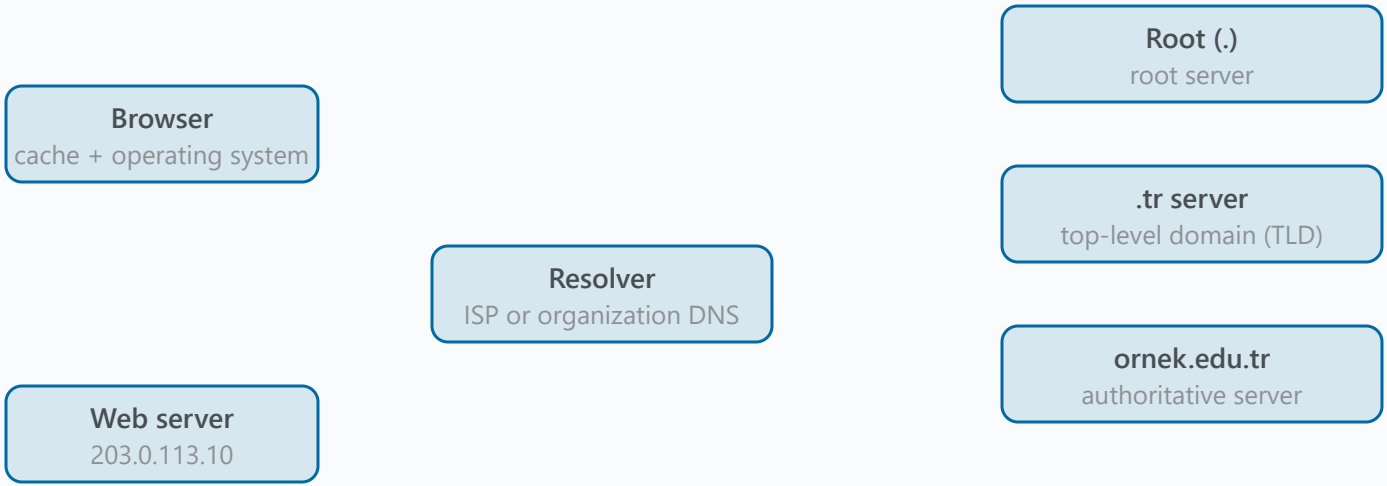
Domain Names and DNS



A domain name is a name people can read. DNS translates it into an IP address you can connect to.

ADDRESSING

DNS Resolution, Step by Step



1. The user types **www.ornek.edu.tr** into the address bar. Connecting requires this name's IP address.

DNS Resolution, Step by Step



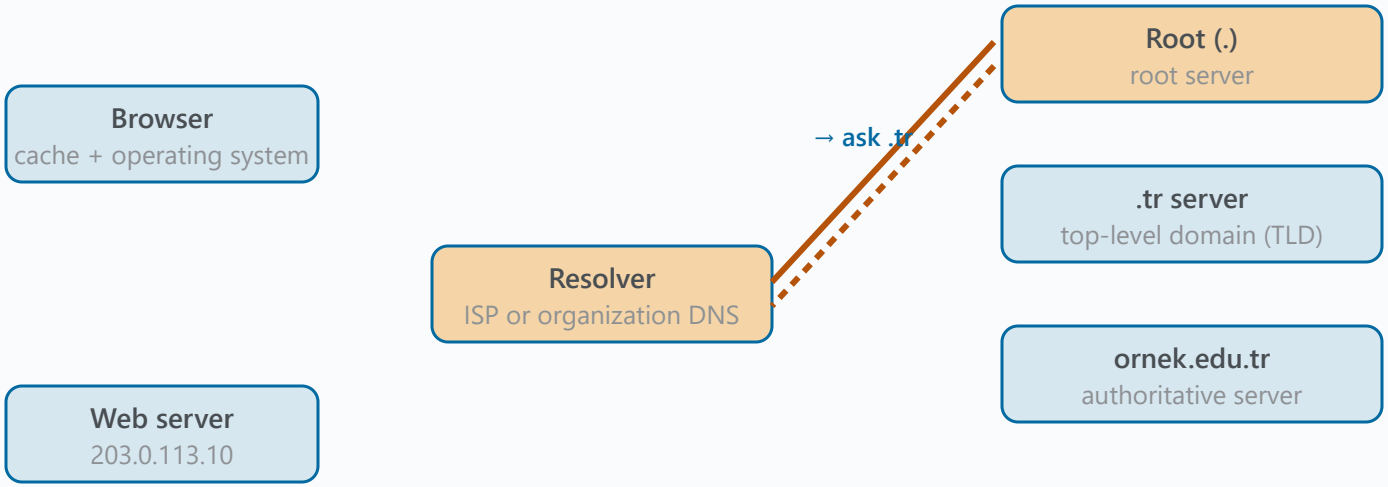
2. The browser and operating system check their own caches first: no record.

DNS Resolution, Step by Step



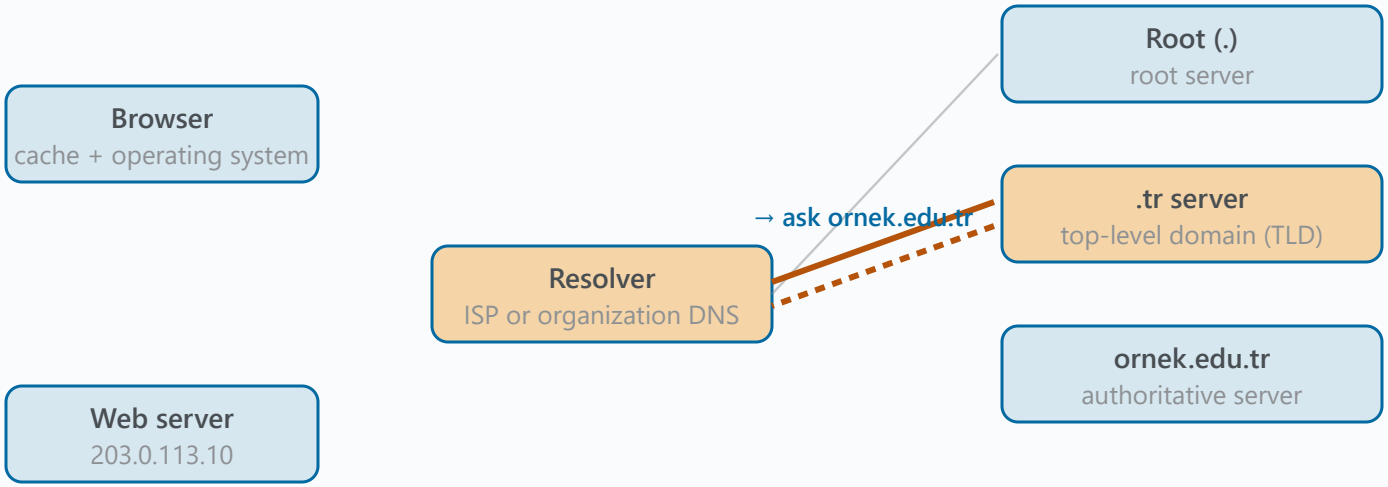
3. The question goes to the DNS resolver in the network settings (usually the ISP's server).

DNS Resolution, Step by Step



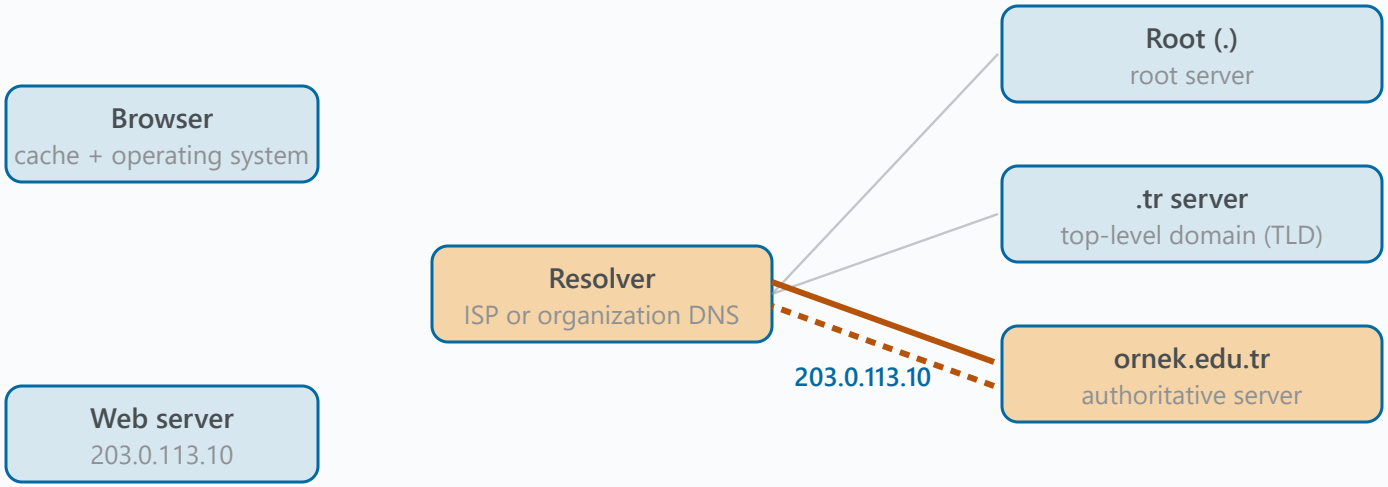
4. The resolver starts at the top: the root server doesn't know the name but says “ask the .tr servers”.

DNS Resolution, Step by Step



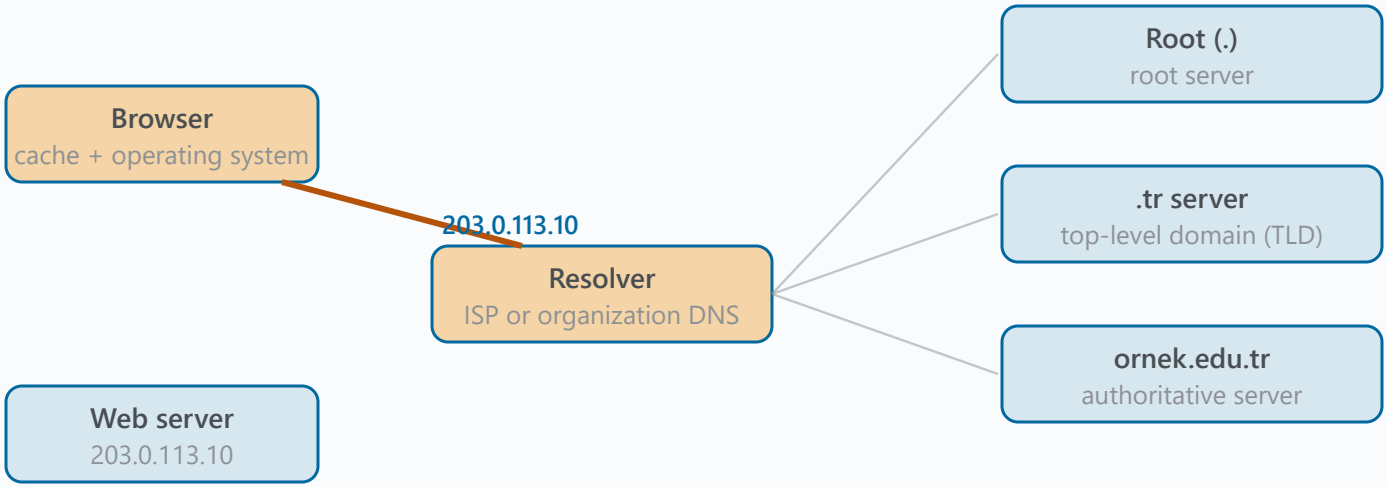
5. The .tr server doesn't know the full name either: “ask the authoritative server for ornek.edu.tr”.

DNS Resolution, Step by Step



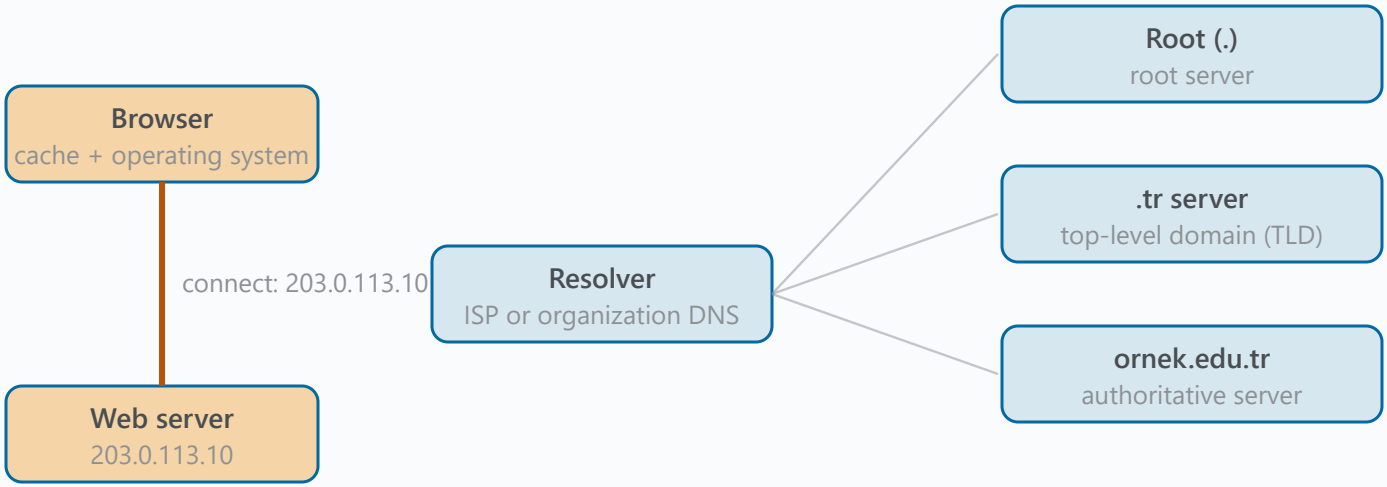
6. The authoritative server knows the record and answers: **203.0.113.10**.

DNS Resolution, Step by Step



7. The resolver passes the answer to the browser and caches it for a while (TTL); the next question is answered at once.

DNS Resolution, Step by Step



8. Now that it knows the IP address, the browser connects to the web server at 203.0.113.10.

The DNS Resolver and Its Cache, Step by Step



resolver cache: empty

The browser asks the resolver for **www.ornek.edu.tr**. The resolver's cache is empty (it just started).

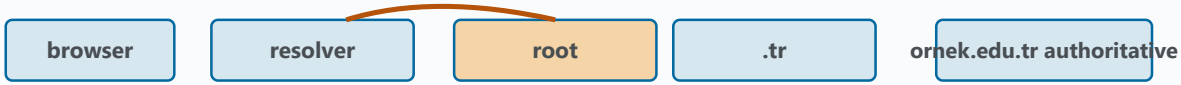
Step 1 / 11

```
▶ is the name in the cache?  Yes → answer, done
2 server = closest known (else root)
3 ask that server for the name
4 referral: cache it, ask the referred server
5 IP received: cache it (TTL), answer
```

VARIABLES

`name = www.ornek.edu.tr` `queries = 0`

The DNS Resolver and Its Cache, Step by Step



resolver cache: empty

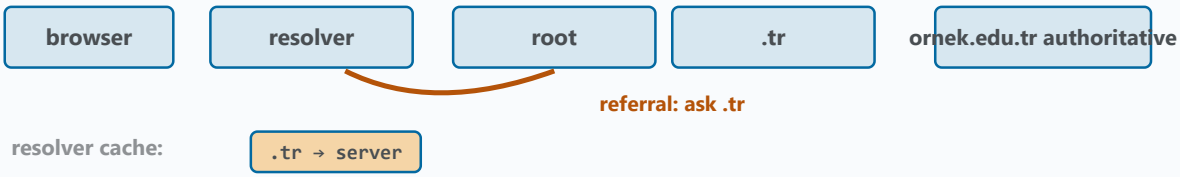
Not in the cache, and no known server either → ask a **root** server.


```
1 is the name in the cache?  Yes → answer, done
2 server = closest known (else root)
▶ ask that server for the name
4 referral: cache it, ask the referred server
5 IP received: cache it (TTL), answer
```

VARIABLES

`name = www.ornek.edu.tr` `queries = 1`

The DNS Resolver and Its Cache, Step by Step



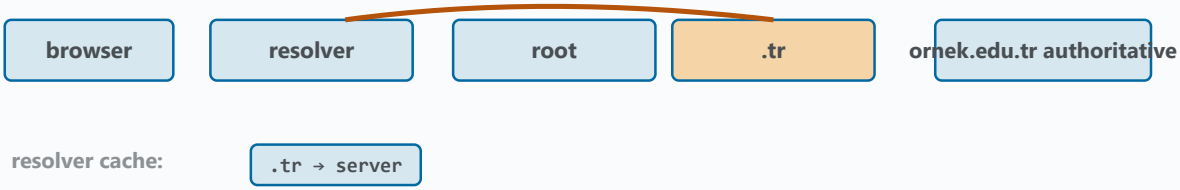
The root doesn't know the name; it points to the **.tr** servers. This is written to the cache.


```
1 is the name in the cache?  Yes → answer, done
2 server = closest known (else root)
3 ask that server for the name
▶ referral: cache it, ask the referred server
5 IP received: cache it (TTL), answer
```

VARIABLES

`name = www.ornek.edu.tr` `queries = 1`

The DNS Resolver and Its Cache, Step by Step



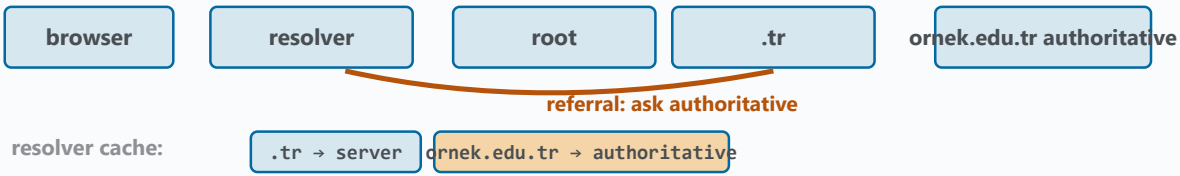
The .tr server is asked.


```
1 is the name in the cache?  Yes → answer, done
2 server = closest known (else root)
▶ ask that server for the name
4 referral: cache it, ask the referred server
5 IP received: cache it (TTL), answer
```

VARIABLES

`name = www.ornek.edu.tr` `queries = 2`

The DNS Resolver and Its Cache, Step by Step



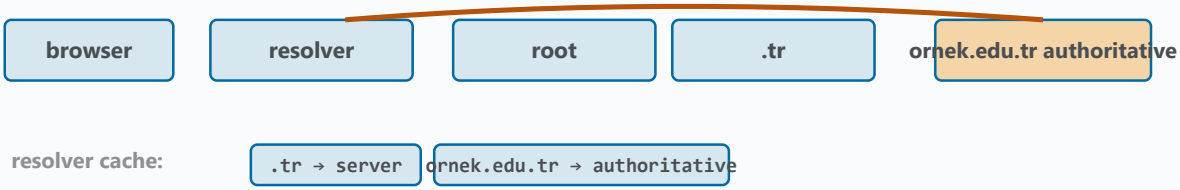
.tr points to the **authoritative server** for ornek.edu.tr; that is cached too.


```
1 is the name in the cache?  Yes → answer, done
2 server = closest known (else root)
3 ask that server for the name
▶ referral: cache it, ask the referred server
5 IP received: cache it (TTL), answer
```

VARIABLES

`name = www.ornek.edu.tr` `queries = 2`

The DNS Resolver and Its Cache, Step by Step



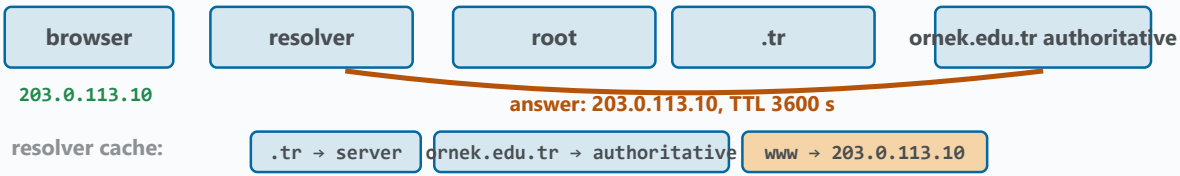
The authoritative server is asked.


```
1 is the name in the cache?  Yes → answer, done
2 server = closest known (else root)
▶ ask that server for the name
4 referral: cache it, ask the referred server
5 IP received: cache it (TTL), answer
```

VARIABLES

`name = www.ornek.edu.tr` `queries = 3`

The DNS Resolver and Its Cache, Step by Step



The authoritative server knows the record: **203.0.113.10**. It is cached for its TTL (here 1 hour) and passed to the browser. 3 queries.

Step 7 / 11

```
1 is the name in the cache?  Yes → answer, done
2 server = closest known (else root)
3 ask that server for the name
4 referral: cache it, ask the referred server
▶ IP received: cache it (TTL), answer
```

VARIABLES

`name = www.ornek.edu.tr` `queries = 3`

`answer = 203.0.113.10`

The DNS Resolver and Its Cache, Step by Step



Within the same minute, another page asks for **ogrenci.ornek.edu.tr**.

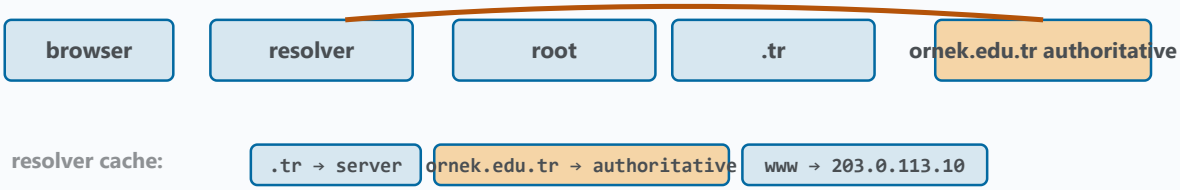

```
▶ is the name in the cache?  Yes → answer, done
2 server = closest known (else root)
3 ask that server for the name
4 referral: cache it, ask the referred server
5 IP received: cache it (TTL), answer
```

VARIABLES

`name = ogrenci.ornek.edu.tr`

`queries = 0`

The DNS Resolver and Its Cache, Step by Step



That name isn't cached, but the authoritative server for ornek.edu.tr is known: root and .tr are **skipped**, and the authoritative server is asked directly.

Step 9 / 11

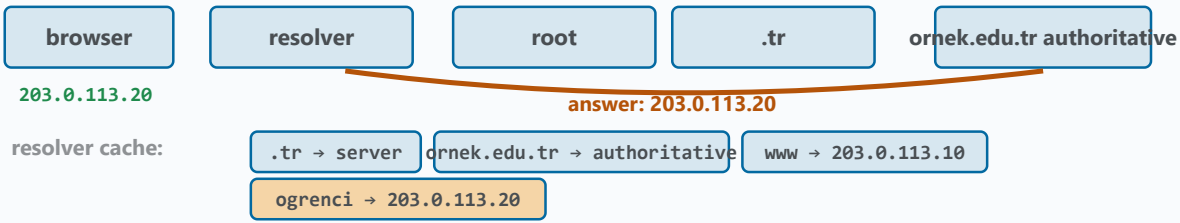
```
1 is the name in the cache?  Yes → answer, done
▶ server = closest known (else root)
3 ask that server for the name
4 referral: cache it, ask the referred server
5 IP received: cache it (TTL), answer
```

VARIABLES

`name = ogrenci.ornek.edu.tr`

`queries = 1`

The DNS Resolver and Its Cache, Step by Step



The answer is **203.0.113.20**; just **1 query**.

Step 10 / 11

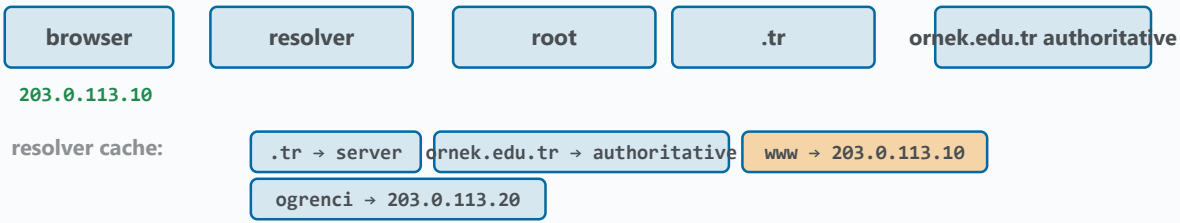
```
1 is the name in the cache?  Yes → answer, done
2 server = closest known (else root)
3 ask that server for the name
4 referral: cache it, ask the referred server
▶ IP received: cache it (TTL), answer
```

VARIABLES

`name = ogrenci.ornek.edu.tr`

`queries = 1` `answer = 203.0.113.20`

The DNS Resolver and Its Cache, Step by Step



Another user asks for **www.ornek.edu.tr**: the record is cached → **0 queries**, an instant answer. When the TTL expires the record is removed and the next question goes back to the authoritative server; that is why an address change doesn't reach everyone at once.

Step 11 / 11

```
▶ is the name in the cache?  Yes → answer, done
2 server = closest known (else root)
3 ask that server for the name
4 referral: cache it, ask the referred server
5 IP received: cache it (TTL), answer
```

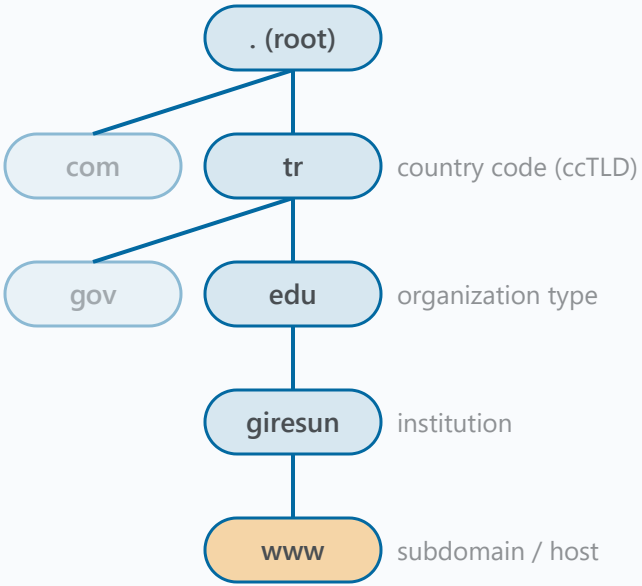
VARIABLES

`name = www.ornek.edu.tr` `queries = 0`

`answer = 203.0.113.10`

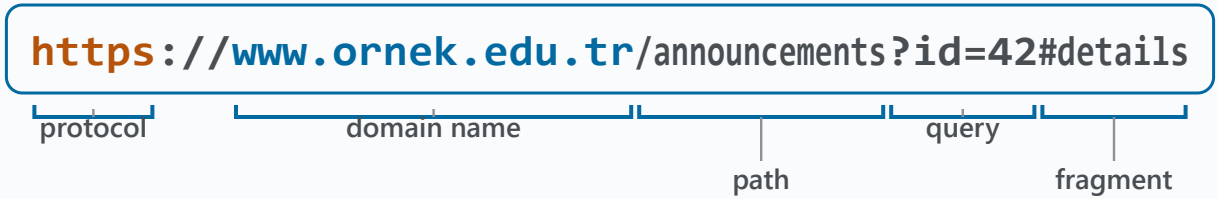
Parts of a Domain Name

www.**giresun**.edu.tr



- A name is read **right to left**: from general to specific.
- **tr:** country-code top-level domain.
- **edu:** organization type (education).
- **giresun:** the institution's domain.
- **www:** subdomain or host name.
- A DNS lookup follows this tree down from the root.

The Structure of a URL

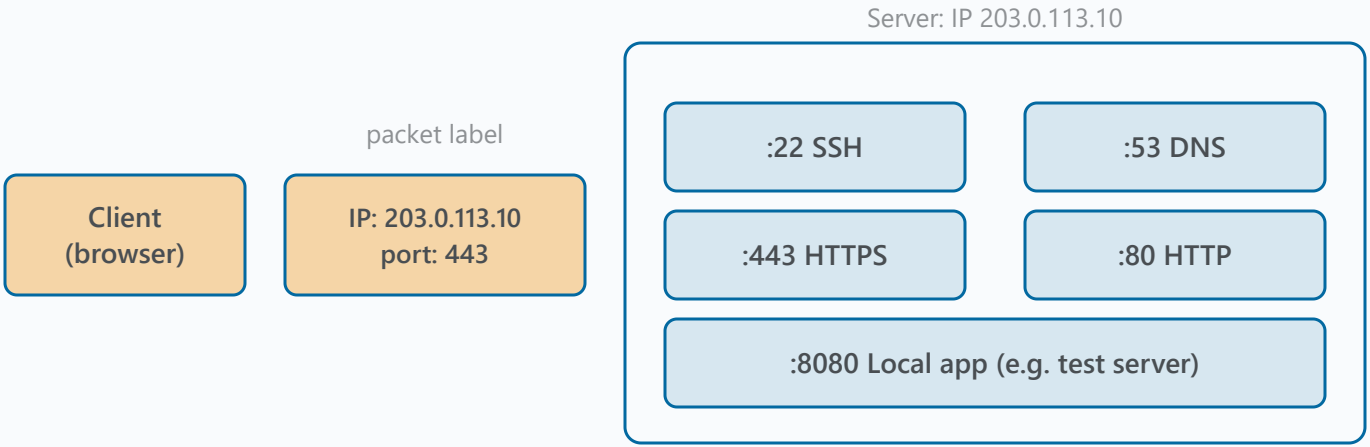


https	The rules for communicating (encrypted HTTP)
www.ornek.edu.tr	The server's domain name; DNS turns it into an IP
/announcements	Path to the resource on the server
?id=42	Extra information for the server (name=value)
#details	A section of the page; not sent to the server, the browser scrolls to it

ADDRESSING

Ports: IP Is the Building, Port Is the Apartment

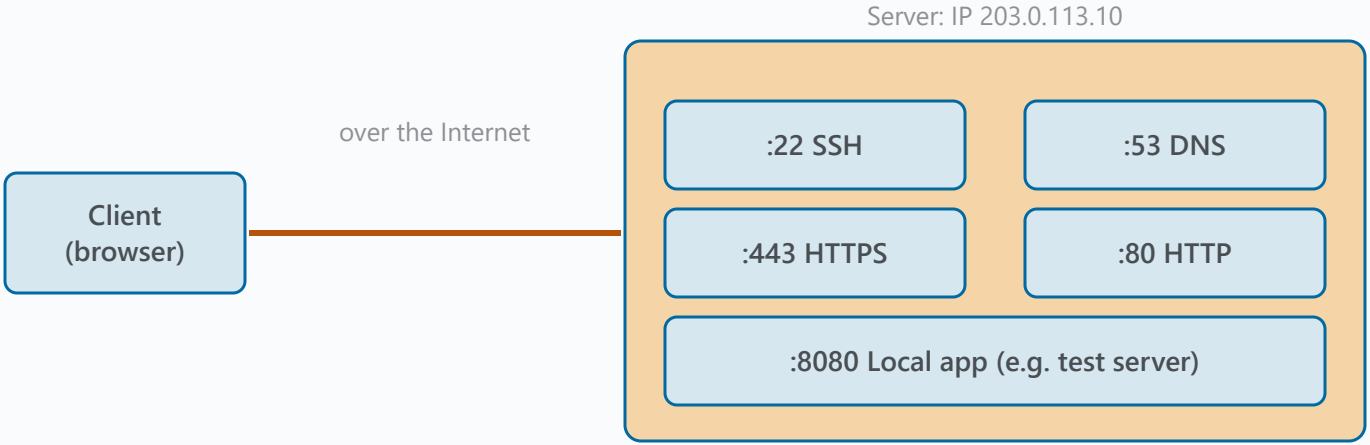
An IP address is like a **building**, a port number like an **apartment** in it. Common ports: **80** HTTP, **443** HTTPS, **22** SSH, **53** DNS.



1. The browser opens **https://ornek.edu.tr**. DNS supplies the IP; the default port for HTTPS is **443**. The packet is addressed to **IP:port**.

Ports: IP Is the Building, Port Is the Apartment

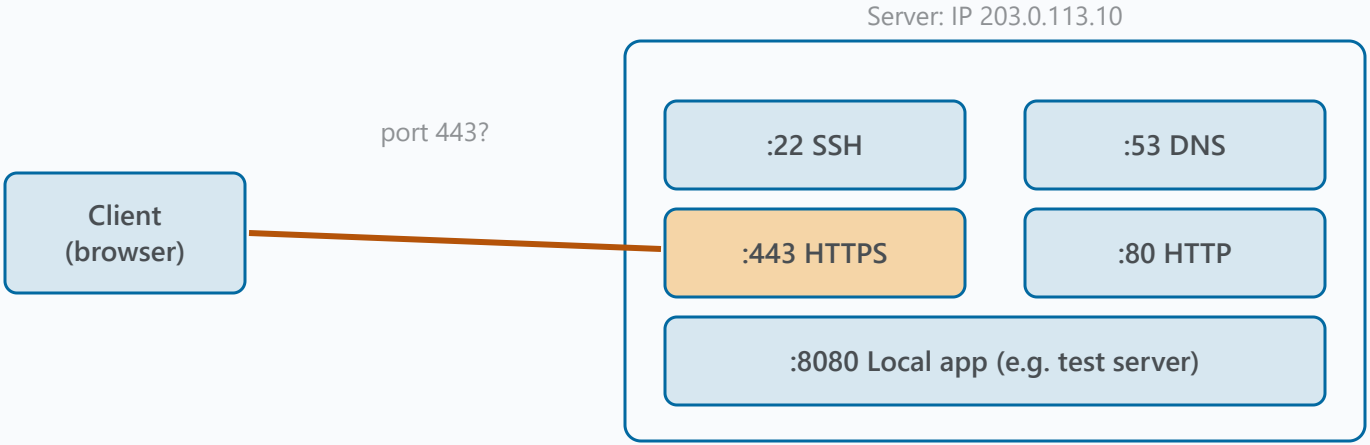
An IP address is like a **building**, a port number like an **apartment** in it. Common ports: **80** HTTP, **443** HTTPS, **22** SSH, **53** DNS.



2. Routers first deliver the packet to the **IP address** (the building). The IP only decides which computer to reach.

Ports: IP Is the Building, Port Is the Apartment

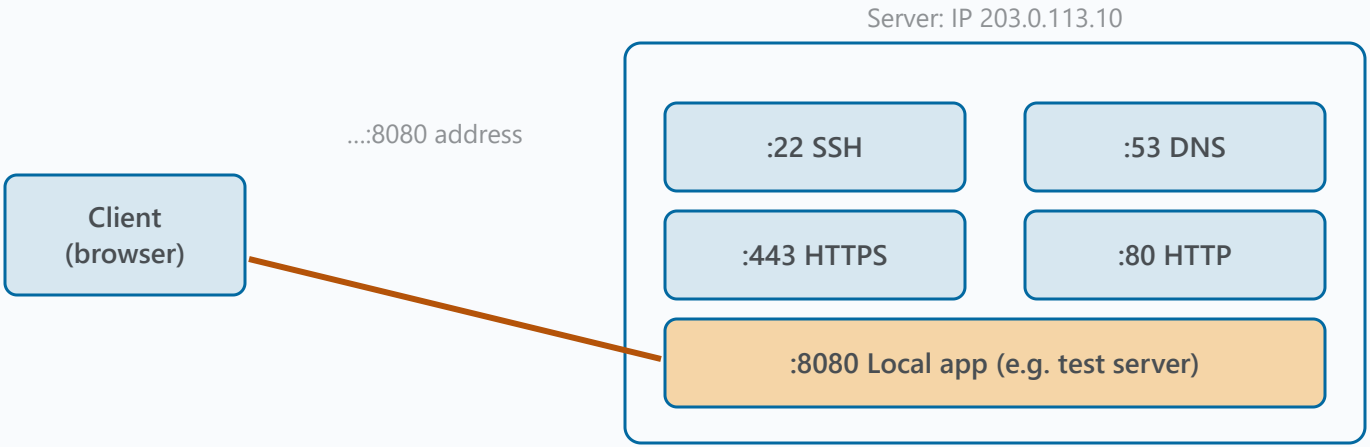
An IP address is like a **building**, a port number like an **apartment** in it. Common ports: **80** HTTP, **443** HTTPS, **22** SSH, **53** DNS.



3. The server's operating system looks at the **port number** (the apartment) and hands the packet to the service listening on it, here the HTTPS web server. :80 and :22 go to other services.

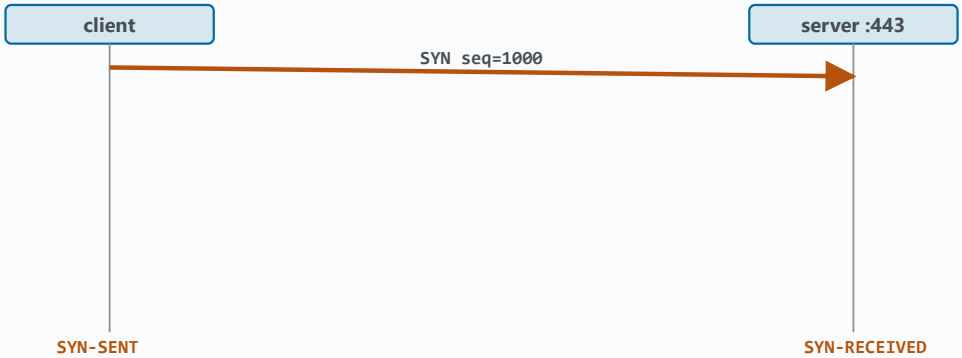
Ports: IP Is the Building, Port Is the Apartment

An IP address is like a **building**, a port number like an **apartment** in it. Common ports: **80** HTTP, **443** HTTPS, **22** SSH, **53** DNS.



4. If the URL includes a port, it replaces the default: **http://203.0.113.10:8080** goes to another application listening on :8080. One IP can host many services on different ports.

TCP: Three-Way Handshake, Data and Teardown



The client asks to connect: a **SYN** segment with initial sequence number 1000 (random in practice). The server was listening on port 443.

Step 1 / 9

```
▶ client → SYN
2 server → SYN + ACK
3 client → ACK        // connected
4 client → data (HTTP request)
5 server → ACK
6 client → FIN;  server → ACK
7 server → FIN;  client → ACK  // closed
```

VARIABLES

`client = SYN-SENT` `server = SYN-RECEIVED`

`client seq = 1000` `server seq = –`

TCP: Three-Way Handshake, Data and Teardown



The server accepts: its own sequence number is 5000 and **ack = 1001**, meaning “got 1000, expecting 1001”. A SYN consumes one sequence number.

Step 2 / 9

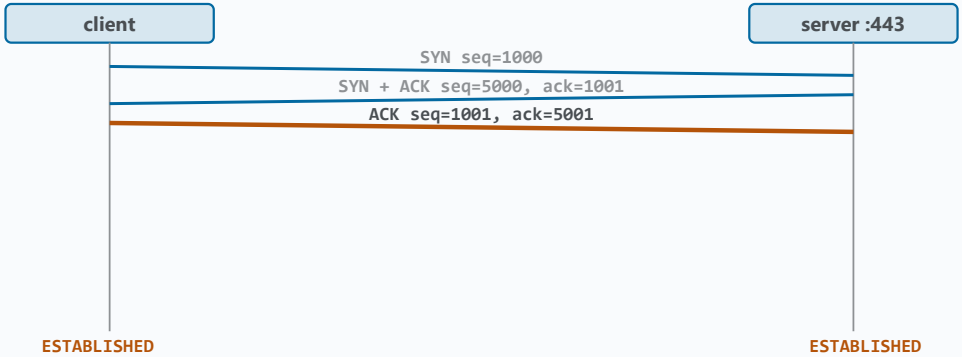
```
1 client → SYN
▶ server → SYN + ACK
3 client → ACK          // connected
4 client → data (HTTP request)
5 server → ACK
6 client → FIN;  server → ACK
7 server → FIN;  client → ACK  // closed
```

VARIABLES

`client = SYN-SENT` `server = SYN-RECEIVED`

`client seq = 1000` `server seq = 5000`

TCP: Three-Way Handshake, Data and Teardown



The client confirms with an **ACK** (ack = 5001). Three segments set up the connection: the **three-way handshake**. No data has been sent yet.

Step 3 / 9

```
1 client → SYN
2 server → SYN + ACK
▶ client → ACK        // connected
4 client → data (HTTP request)
5 server → ACK
6 client → FIN;  server → ACK
7 server → FIN;  client → ACK  // closed
```

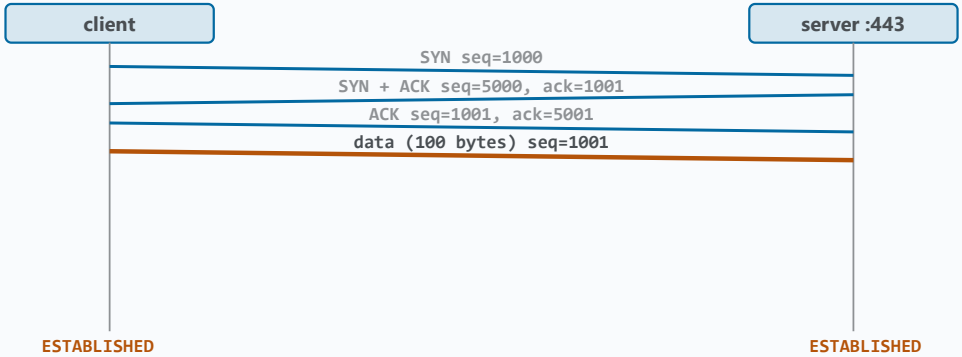
VARIABLES

`client = ESTABLISHED`

`server = ESTABLISHED` `client seq = 1001`

`server seq = 5001`

TCP: Three-Way Handshake, Data and Teardown



The client sends a 100-byte HTTP request: bytes 1001–1100. With HTTPS, a TLS handshake also runs over this channel first.

Step 4 / 9

```
1 client → SYN
2 server → SYN + ACK
3 client → ACK        // connected
▶ client → data (HTTP request)
5 server → ACK
6 client → FIN;  server → ACK
7 server → FIN;  client → ACK  // closed
```

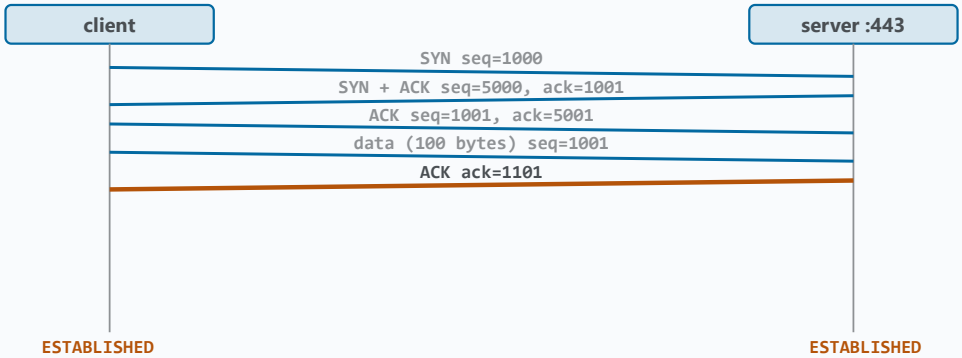
VARIABLES

`client = ESTABLISHED`

`server = ESTABLISHED` `client seq = 1101`

`server seq = 5001`

TCP: Three-Way Handshake, Data and Teardown



With **ack = 1101**, the server reports that all 100 bytes arrived. Without the ACK, the client would resend the same bytes after a timeout.

Step 5 / 9

```
1 client → SYN
2 server → SYN + ACK
3 client → ACK        // connected
4 client → data (HTTP request)
▶ server → ACK
6 client → FIN;  server → ACK
7 server → FIN;  client → ACK  // closed
```

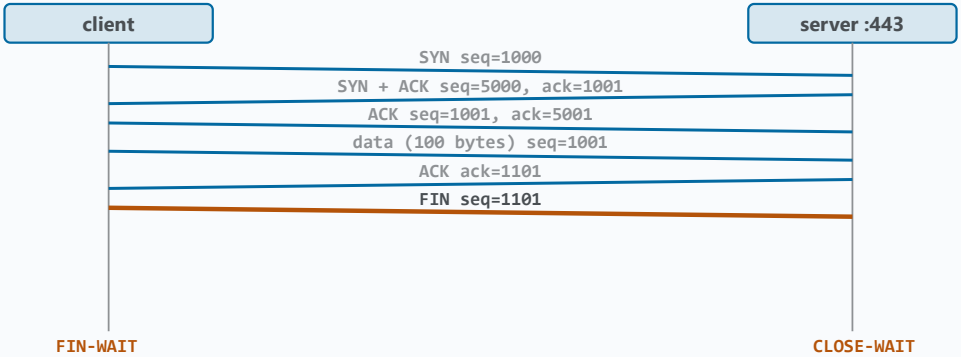
VARIABLES

`client = ESTABLISHED`

`server = ESTABLISHED` `client seq = 1101`

`server seq = 5001`

TCP: Three-Way Handshake, Data and Teardown



Done; the client closes its direction with a **FIN**.

Step 6 / 9

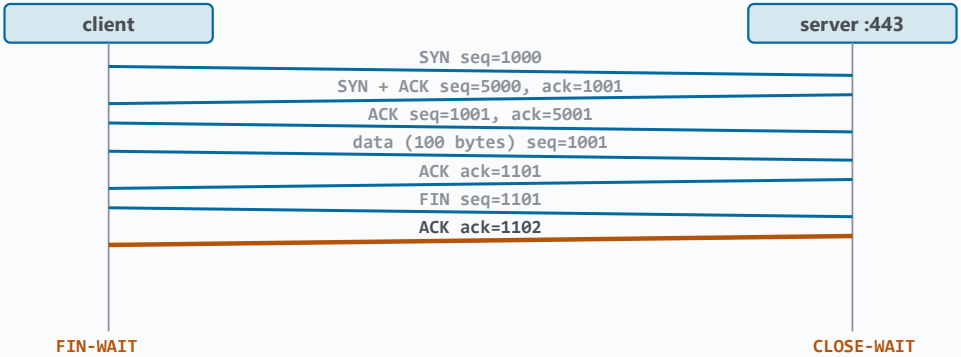
```
1 client → SYN
2 server → SYN + ACK
3 client → ACK        // connected
4 client → data (HTTP request)
5 server → ACK
▶ client → FIN;  server → ACK
7 server → FIN;  client → ACK  // closed
```

VARIABLES

`client = FIN-WAIT` `server = CLOSE-WAIT`

`client seq = 1102` `server seq = 5001`

TCP: Three-Way Handshake, Data and Teardown



The server acknowledges the FIN (ack = 1102). If it still has data to send, it can keep sending.

Step 7 / 9

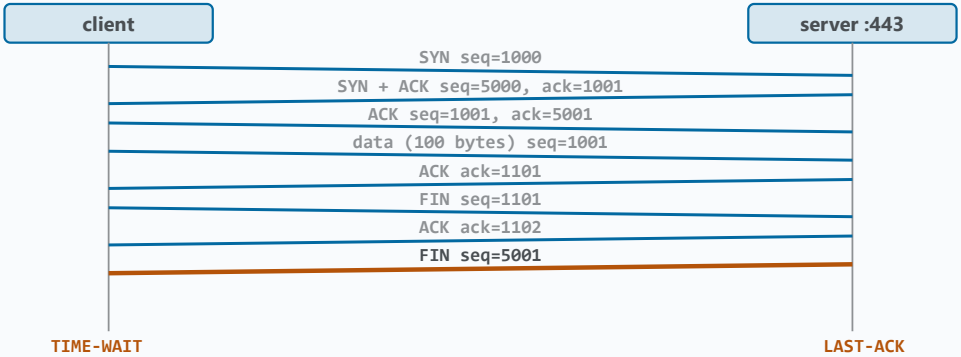
```
1 client → SYN
2 server → SYN + ACK
3 client → ACK        // connected
4 client → data (HTTP request)
5 server → ACK
▶ client → FIN;  server → ACK
7 server → FIN;  client → ACK  // closed
```

VARIABLES

`client = FIN-WAIT` `server = CLOSE-WAIT`

`client seq = 1102` `server seq = 5001`

TCP: Three-Way Handshake, Data and Teardown



The server sends its own **FIN**.

Step 8 / 9

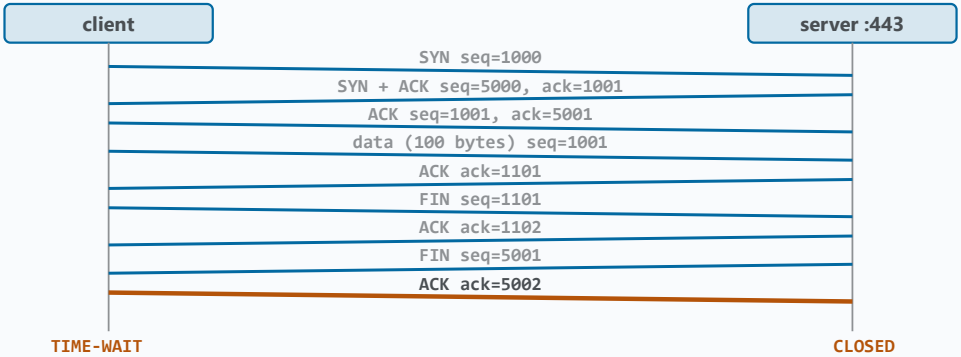
```
1 client → SYN
2 server → SYN + ACK
3 client → ACK        // connected
4 client → data (HTTP request)
5 server → ACK
6 client → FIN;  server → ACK
▶ server → FIN;  client → ACK  // closed
```

VARIABLES

`client = TIME-WAIT` `server = LAST-ACK`

`client seq = 1102` `server seq = 5002`

TCP: Three-Way Handshake, Data and Teardown



The client sends the final **ACK**; the connection is closed in both directions. Setup took 3 segments, teardown 4. UDP has none of these steps.

Step 9 / 9

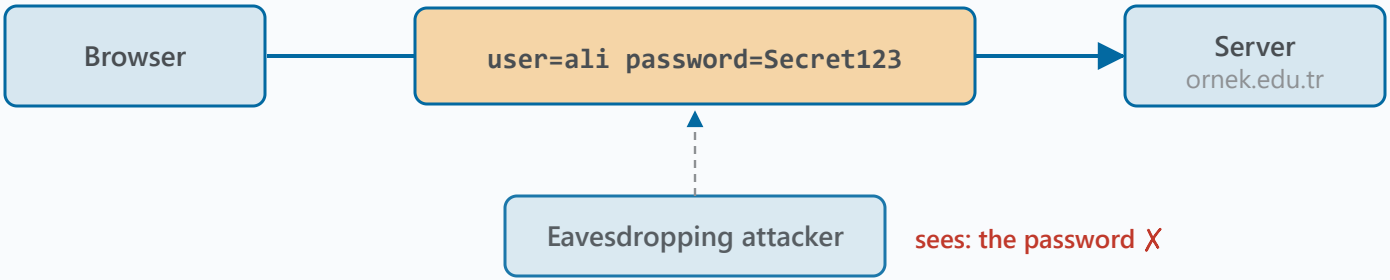
```
1 client → SYN
2 server → SYN + ACK
3 client → ACK        // connected
4 client → data (HTTP request)
5 server → ACK
6 client → FIN;  server → ACK
▶ server → FIN;  client → ACK  // closed
```

VARIABLES

`client = TIME-WAIT` `server = CLOSED`

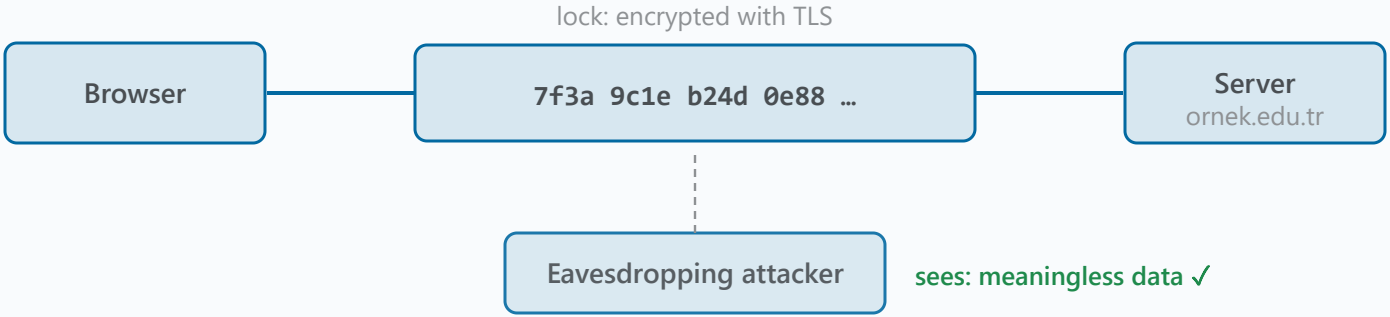
`client seq = 1102` `server seq = 5002`

HTTP vs. HTTPS: What Does Encryption Change?



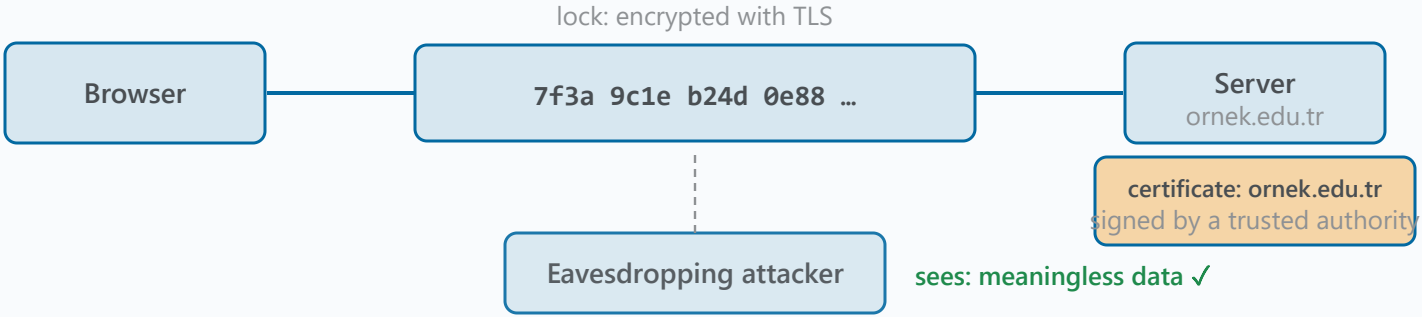
HTTP: If you log in over HTTP on public Wi-Fi, someone on the same network can read the packets and see the password in plain text.

HTTP vs. HTTPS: What Does Encryption Change?



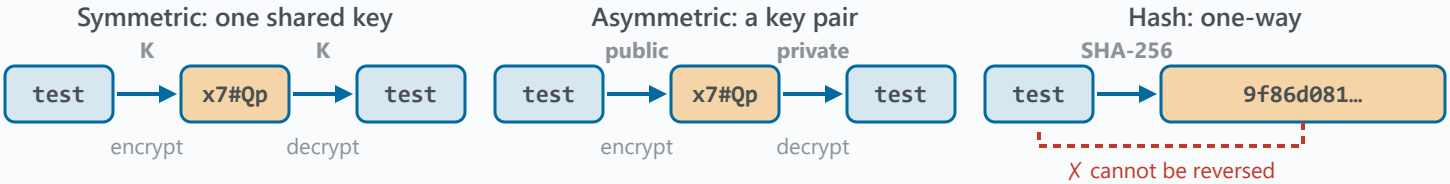
HTTPS: Data is encrypted with TLS: an interceptor sees only meaningless bytes. The browser's “connection is secure” indicator in the address bar shows this (a padlock in older versions).

HTTP vs. HTTPS: What Does Encryption Change?



Certificate: The browser also checks the server's **certificate**: it verifies that you really reached the organization behind ornek.edu.tr. The padlock doesn't mean the site is honest; phishing sites use HTTPS too.

Encryption Basics



Symmetric encryption

The **same key** encrypts and decrypts. It is fast; the problem is getting the key to the other side safely. Example: AES.

Asymmetric encryption

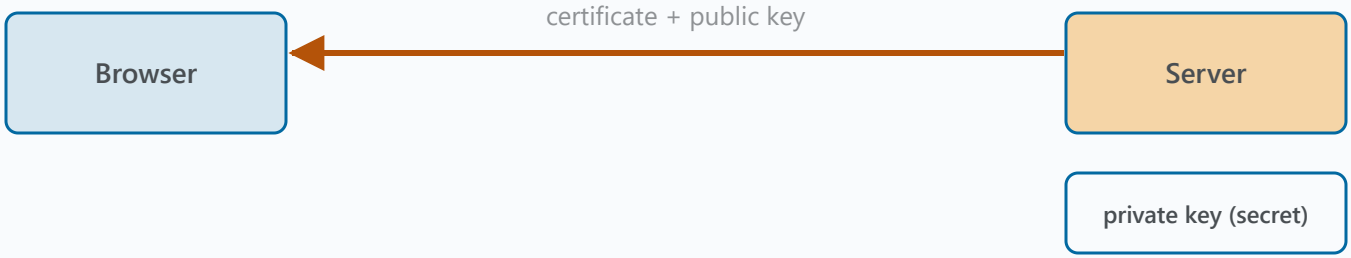
The **public key** is given to everyone (encrypts/verifies); the **private key** stays secret (decrypts/signs). Slower, but it solves key distribution. Examples: RSA, elliptic curves.

Hash

Produces a fixed-length fingerprint of the data. **It is not encryption:** it cannot be reversed; used for storing passwords and checking integrity. Example: SHA-256.

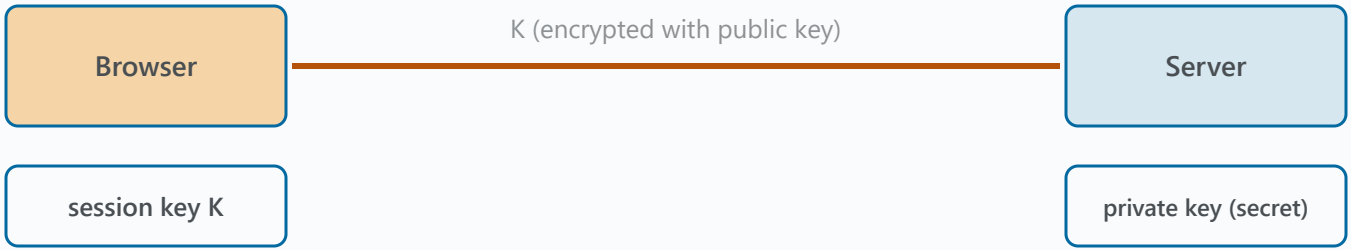
HTTPS uses both **together**: asymmetric for agreeing on a key, symmetric for the actual data. A hash is no substitute for encryption; “encrypted” and “hashed” are different things.

How Encryption Works Together in HTTPS



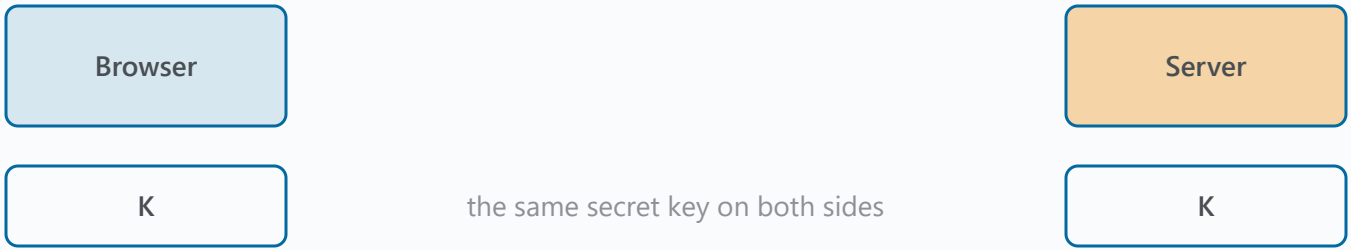
1. The server sends the browser its **certificate and public key**. The private key stays secret on the server.

How Encryption Works Together in HTTPS



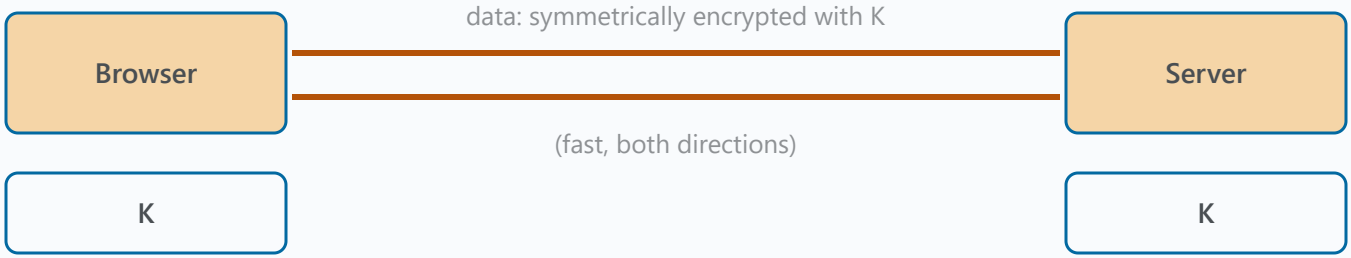
2. The browser generates a random **session key (K)**, encrypts it with the server's public key and sends it (the asymmetric step). An eavesdropper can't decrypt K. This is the older “RSA key transport” model. In TLS 1.3 both sides derive a shared key with Diffie–Hellman; the certificate's public key only proves the server's identity (with a signature).

How Encryption Works Together in HTTPS



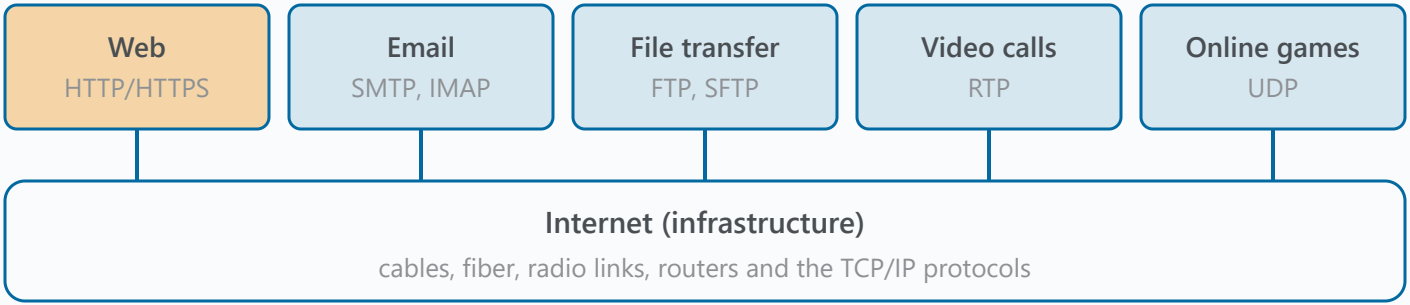
3. The server decrypts K with its **private key**. Both sides now share the same secret key; asymmetric encryption has done its job.

How Encryption Works Together in HTTPS



4. The actual data (page, form, password) flows both ways **encrypted symmetrically with K**: symmetric encryption is fast, and asymmetric is used only to deliver the key safely.

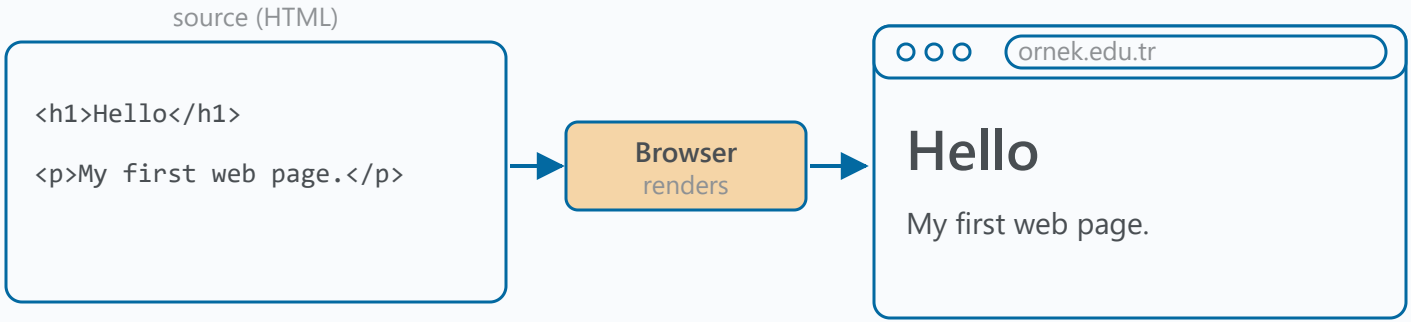
The Internet and the Web



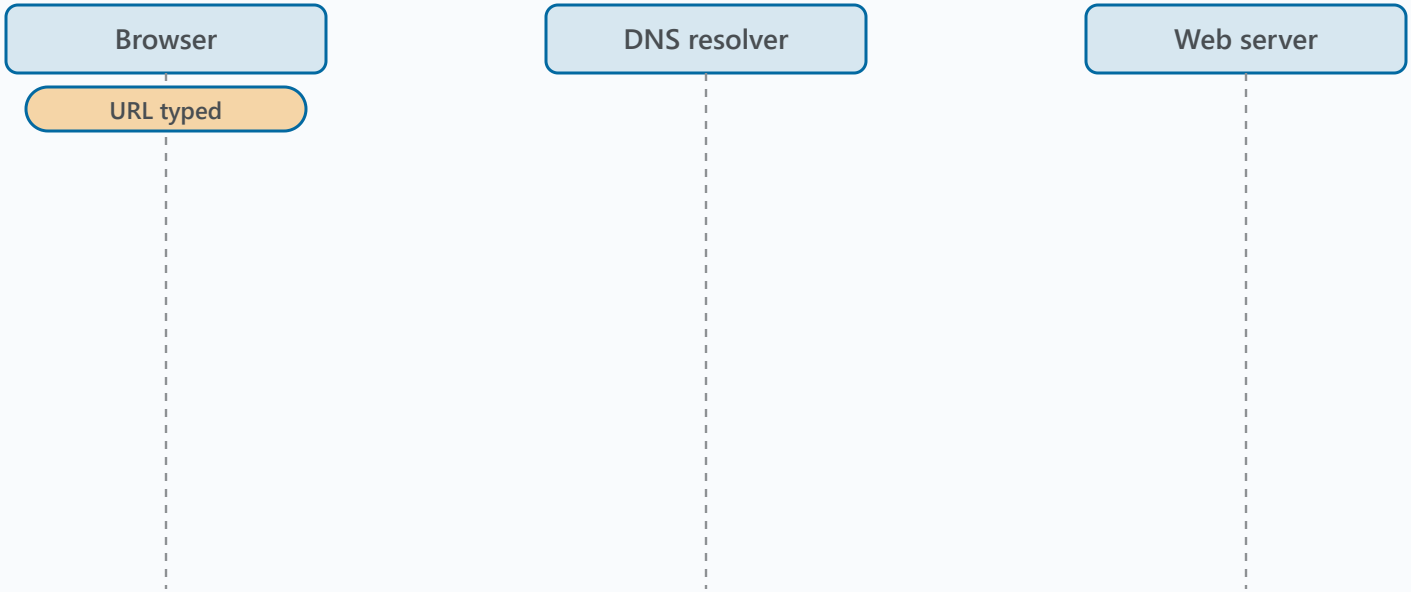
The Web is not the whole Internet. The Internet is the infrastructure; the Web is one of the services on top of it (linked pages and web applications).

HTML and the Web Browser

- HTML defines the structure of a page.
- CSS controls its appearance.
- JavaScript adds interaction and behavior.
- The browser interprets these resources and displays them.

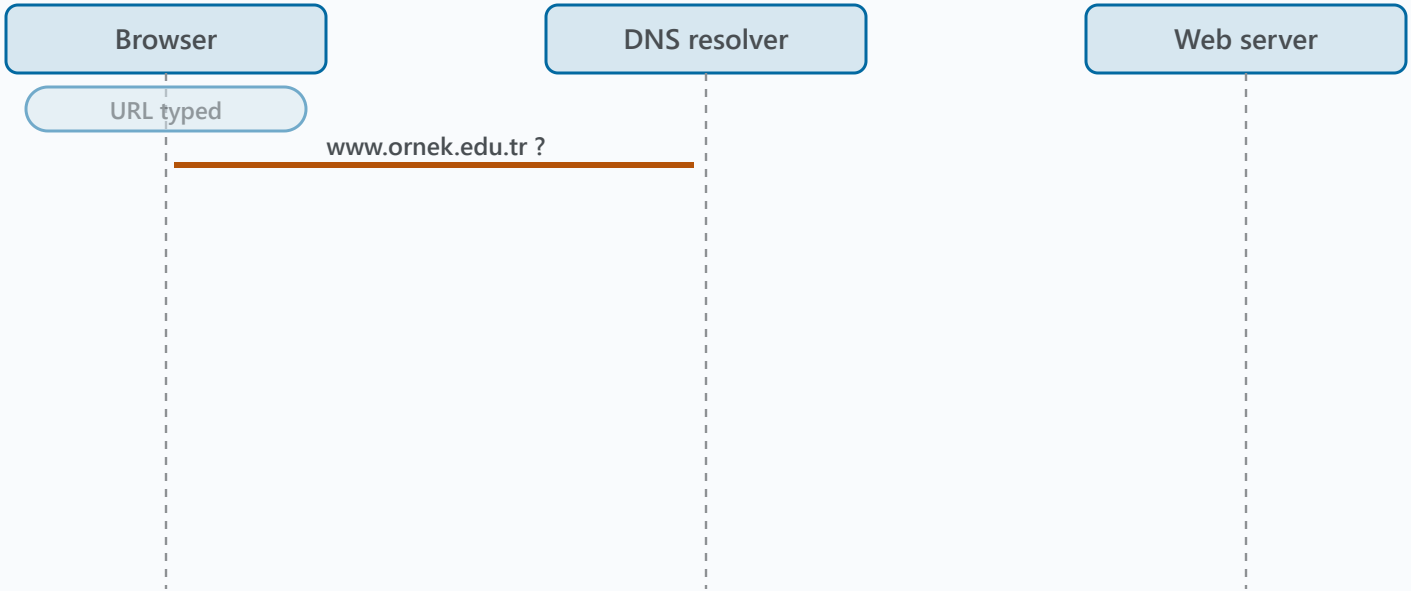


The Journey of a Web Request



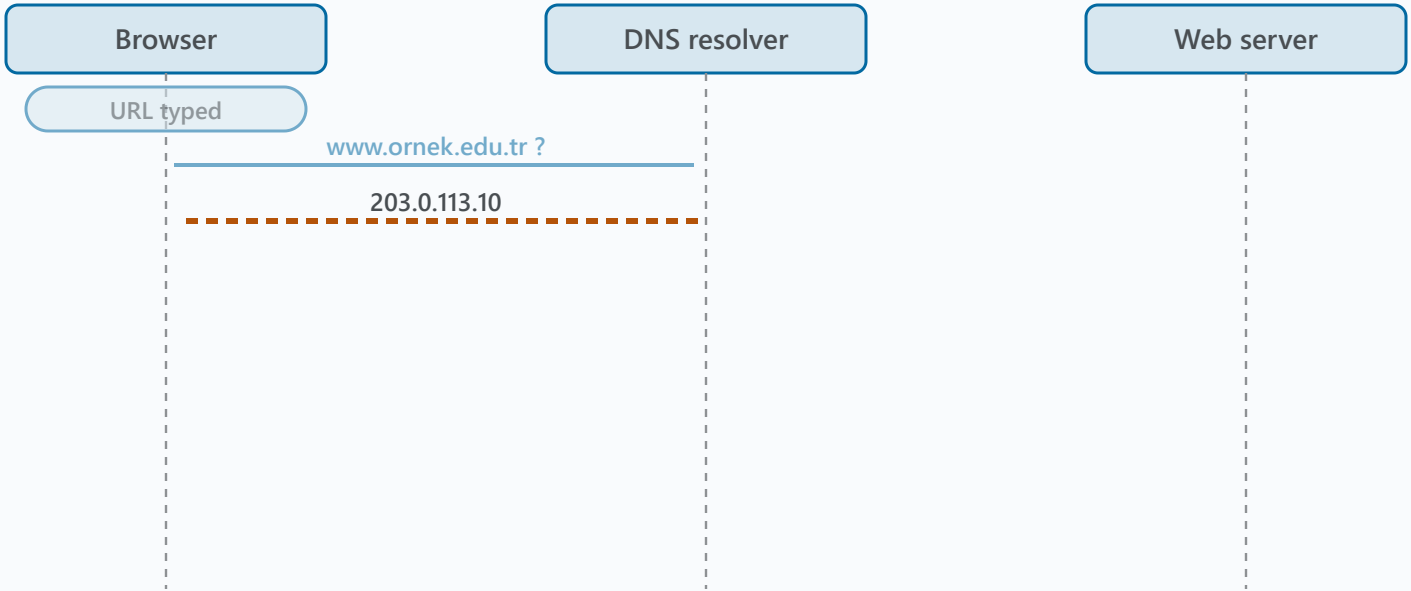
1. The user types **https://www.ornek.edu.tr/announcements** and presses Enter; the browser splits the URL into its parts.

The Journey of a Web Request



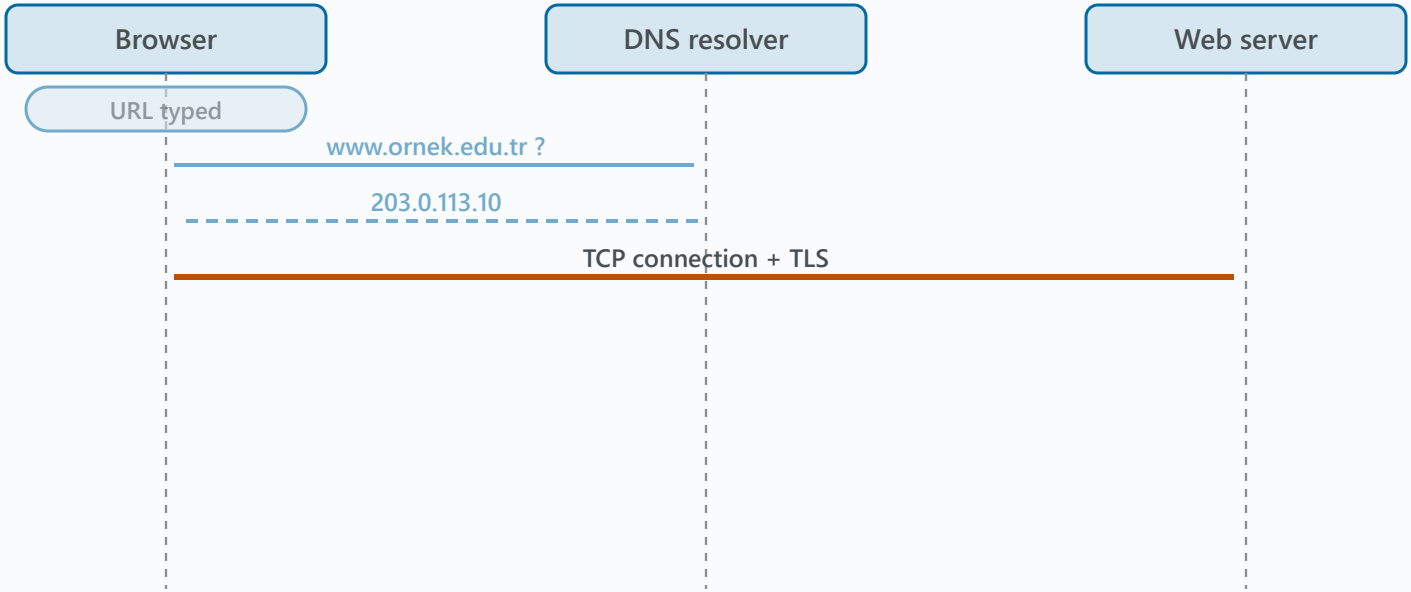
2. The browser asks for the domain's IP address (DNS resolution).

The Journey of a Web Request



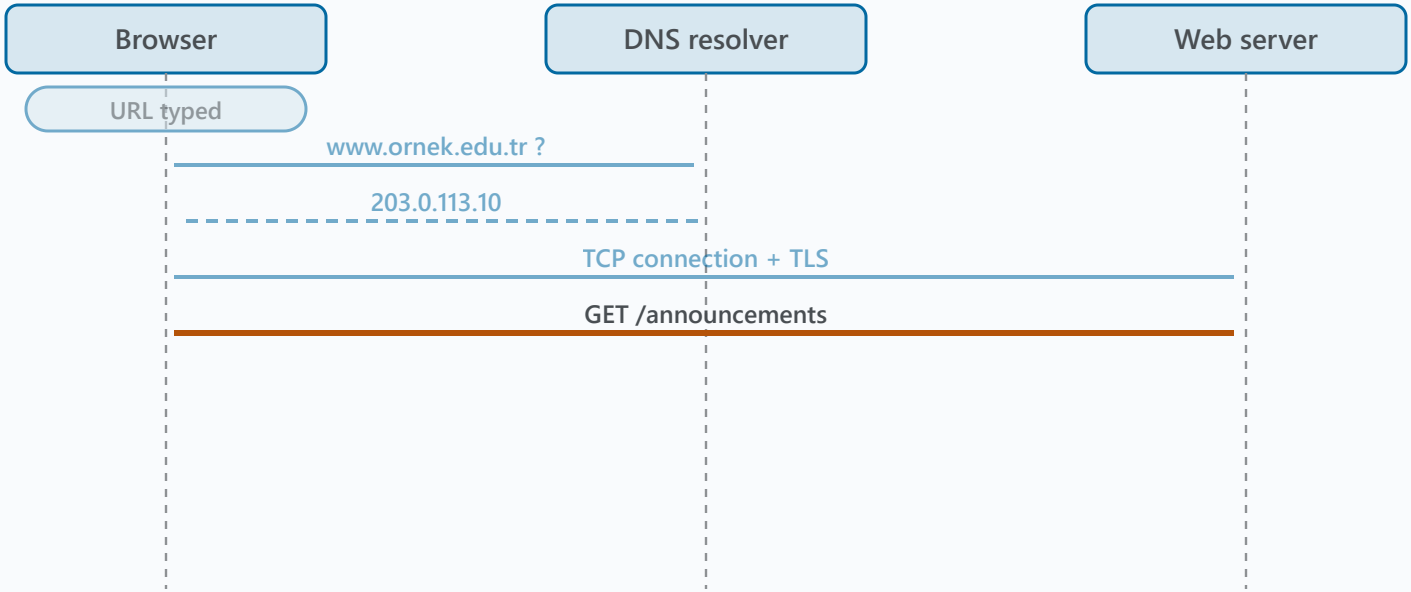
3. The answer arrives: the server is at 203.0.113.10.

The Journey of a Web Request



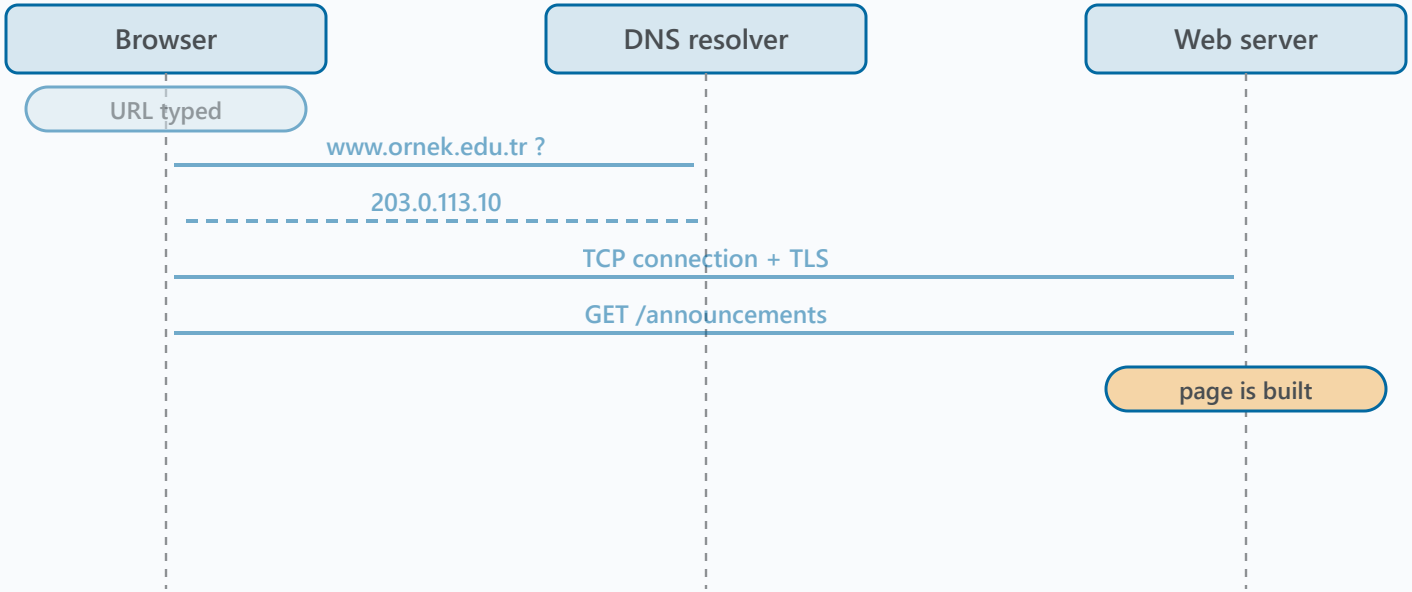
4. The browser opens a TCP connection to the server; since this is HTTPS, the two sides agree on encryption keys with TLS.

The Journey of a Web Request



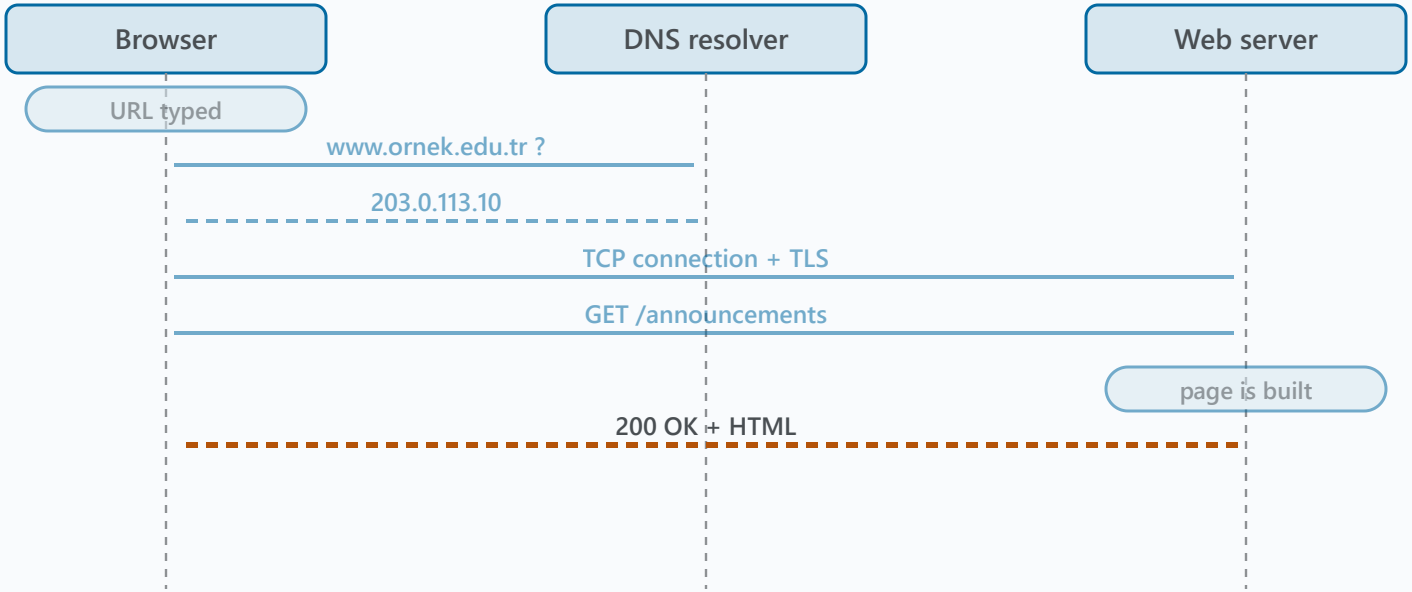
5. The HTTP request goes over the encrypted channel: “give me /announcements”.

The Journey of a Web Request



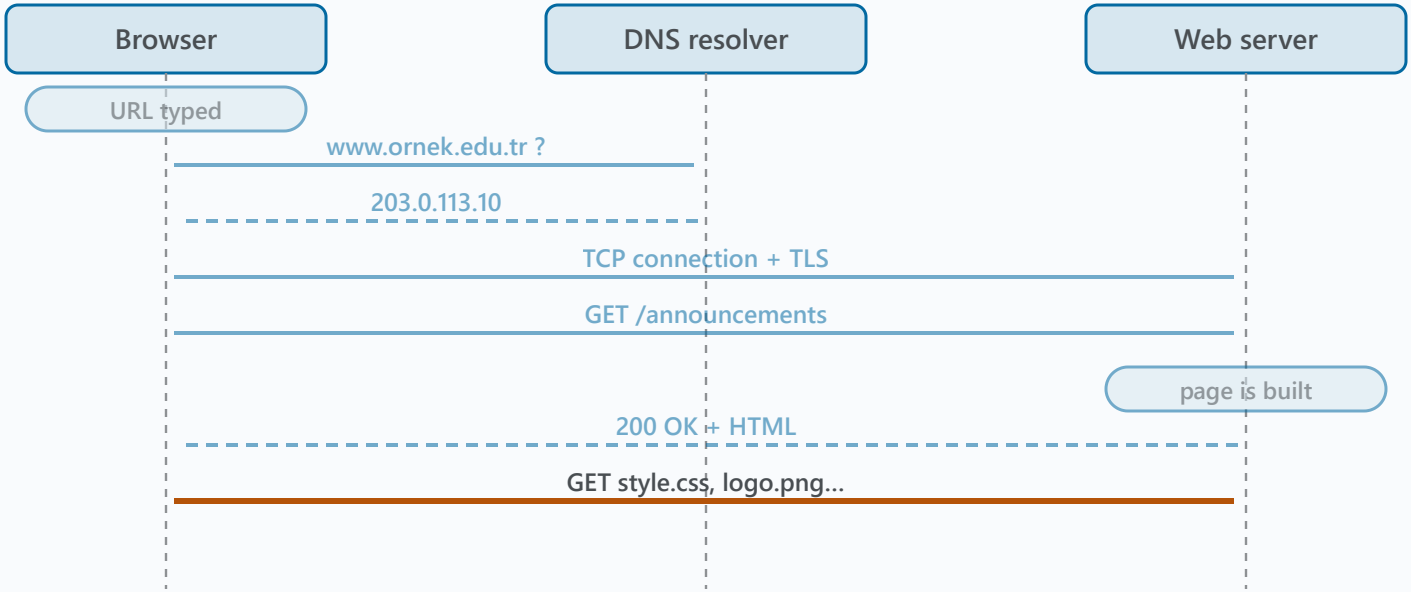
6. The server handles the request: it reads a file, or pulls the announcements from a database and generates HTML.

The Journey of a Web Request



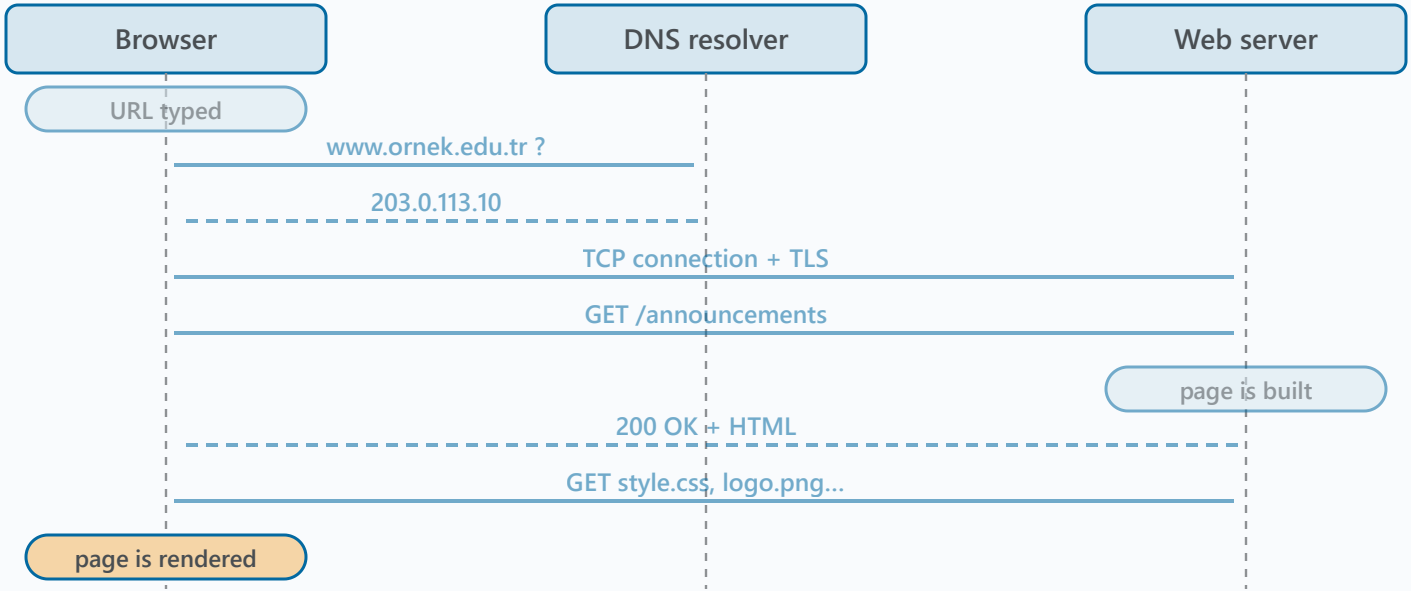
7. The response comes back: status code **200 OK** (success) and the page's HTML. (404 if not found.)

The Journey of a Web Request



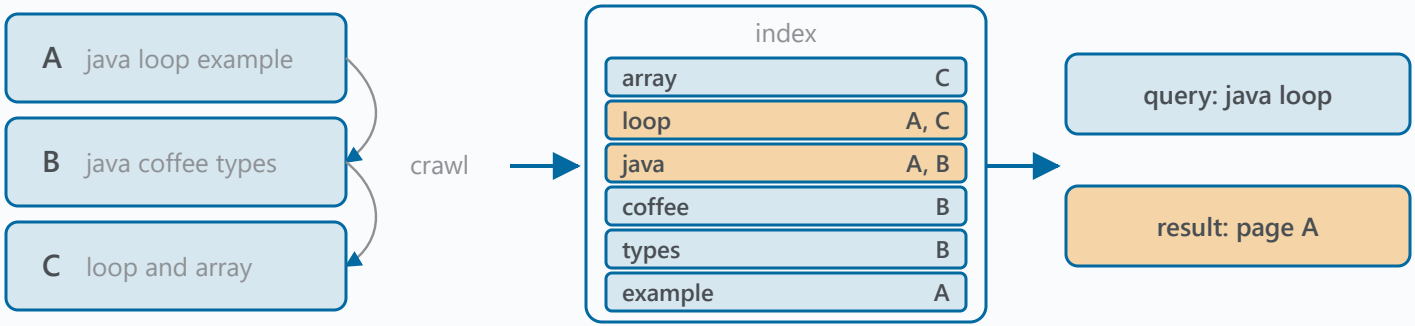
8. While reading the HTML, the browser fetches the CSS, JavaScript and images it needs with separate requests.

The Journey of a Web Request



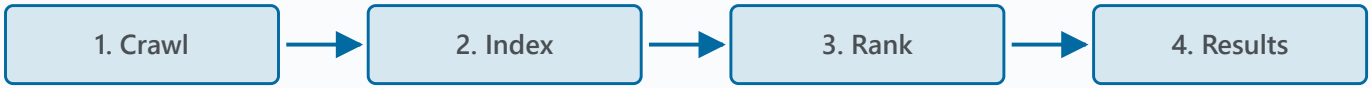
9. The browser combines the HTML structure, CSS styling and JavaScript behavior and draws the page on screen.

How Does a Search Engine Work?



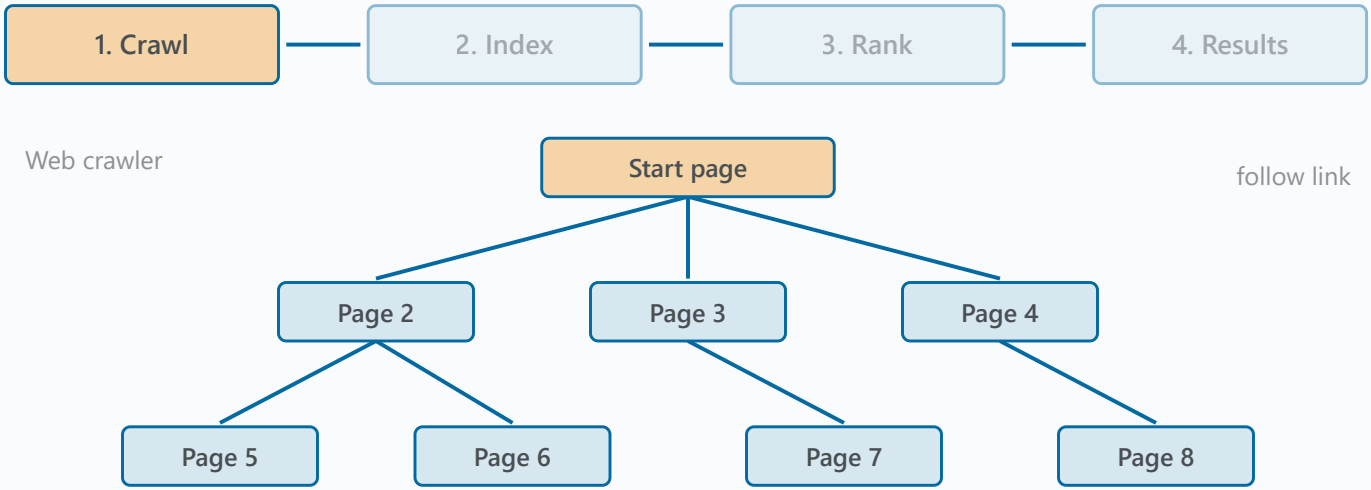
- Web crawlers discover pages by following links.
- The content is stored in a search index.
- A query is answered by intersecting the pages that contain each word in the index (the Web itself is not searched).
- Results are ranked by relevance, freshness, links and other signals.

Search Engines: Crawl, Index, Rank



When you search, the Web is not visited at that moment; the answer comes from an index **built in advance**. Four stages: crawl, index, rank, results.

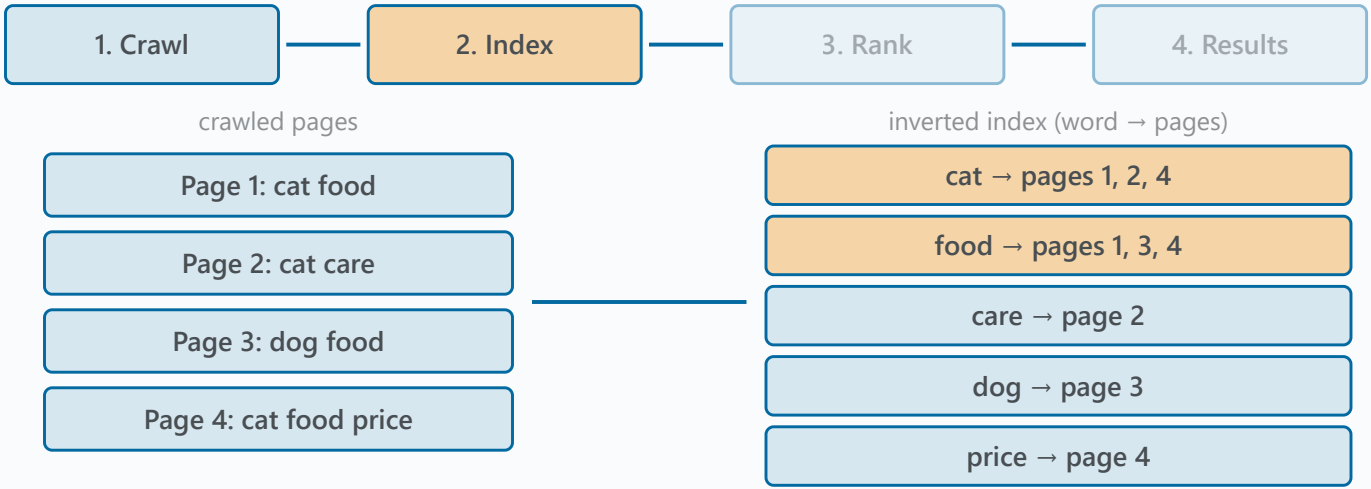
Search Engines: Crawl, Index, Rank



Every link found is queued, and the page content is downloaded.

Crawl: the crawler follows links from seed pages to discover new pages and download their content. Only reachable, permitted pages are crawled, not the whole Web.

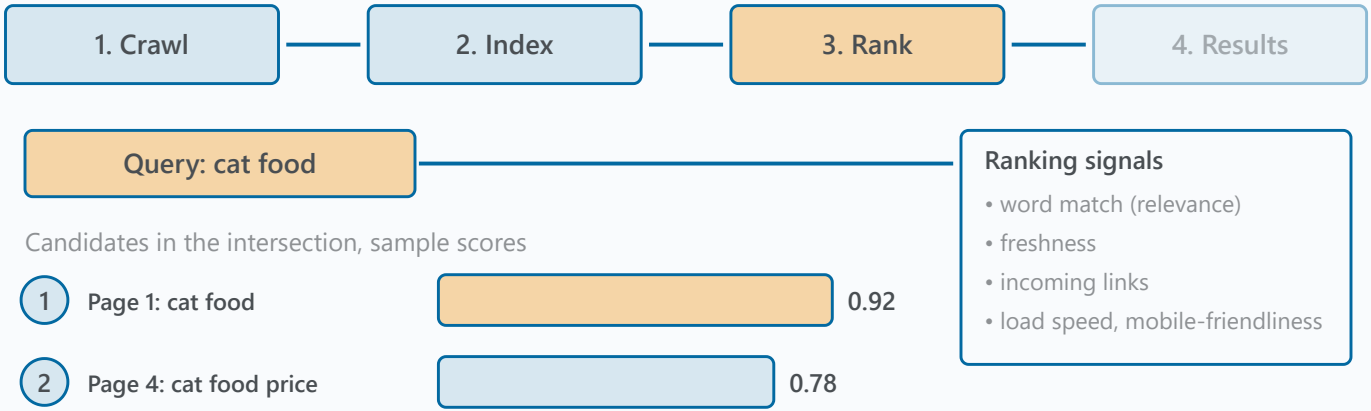
Search Engines: Crawl, Index, Rank



"cat food": {1, 2, 4} ∩ {1, 3, 4} = {1, 4}; two candidate pages remain.

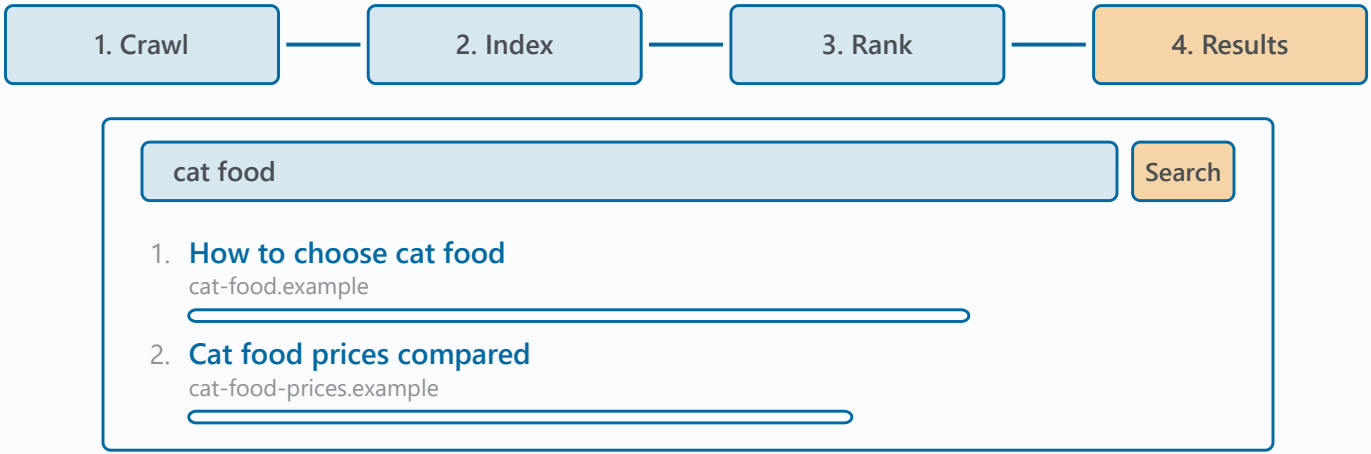
Index: the words on each page go into an **inverted index** (word → pages). At query time, pages aren't read one by one; the lists are consulted.

Search Engines: Crawl, Index, Rank



Rank: the candidates containing both words (the intersection: Page 1 and Page 4) are scored and ordered by relevance, freshness, incoming links and so on (the scores are just examples).

Search Engines: Crawl, Index, Rank



Results: the top-scoring pages are listed with title, address and snippet. The whole process usually takes a fraction of a second.

Effective Searching

Technique	Example	Purpose
Exact phrase	`"computational thinking"`	Search for the words together
Exclusion	`java -coffee`	Remove an unwanted meaning
Site restriction	`site:gov.tr scholarship`	Search within certain domains
File type	`filetype:pdf network security`	Restrict the document format

Evaluating Sources and AI Output

Question	What to look for
Author	Who wrote it? An expert or an accountable organization?
Date	When was it published or updated? Is it current for the topic?
Source	Where did the information come from? Are citations and links given?
Verification	Can the data and claims be checked?
Cross-checking	Do other independent, reliable sources say the same?

AI output can be wrong too: fluent, confident text may contain false or made-up information (and sources). Check important facts against the criteria above.

DIGITAL SERVICES

Online Services

Government

e-Government, health, tax and education services.

Communication

Social networks, instant messaging and community platforms.

Media

Video, music and digital broadcasting.

Commerce

Stores, marketplaces and payment systems.

Cloud

File storage, collaboration and applications.

AI

Tools for text, images and decision support.

DIGITAL SERVICES

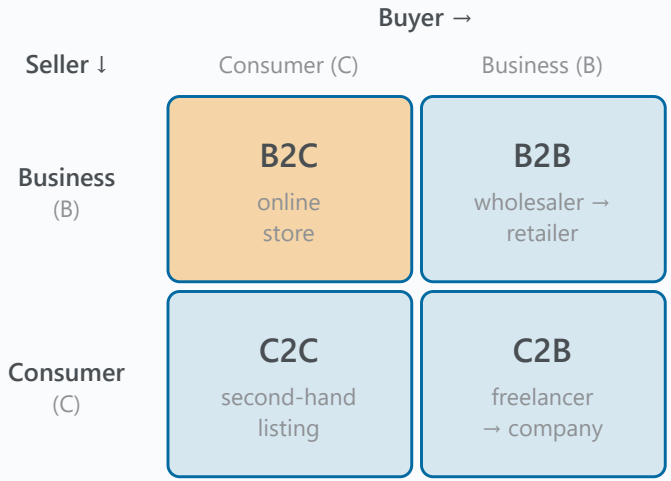
Cloud Service Models

Layer	On-premises	IaaS	PaaS	SaaS
Application	You	You	You	Provider
Data	You	You	You	You (content)
Runtime	You	You	Provider	Provider
Operating system	You	You	Provider	Provider
Server and network hardware	You	Provider	Provider	Provider
Example	Your own server room	Renting virtual servers	App hosting platform	Email, online office

A **CDN** (content delivery network) keeps copies of the same content on servers around the world and serves each user from a nearby one, cutting latency.

DIGITAL SERVICES

E-Commerce



B2C
Firms sell to consumers.

B2B
Trade between businesses.

C2C
Consumers sell to each other.

C2B
An individual offers goods or services to a business.

Check the seller, price, delivery, return policy and payment security before you buy.

DIGITAL SERVICES

Email Addresses

sercan.kulcu@giresun.edu.tr

User name

Identifies the mailbox.

The @ sign

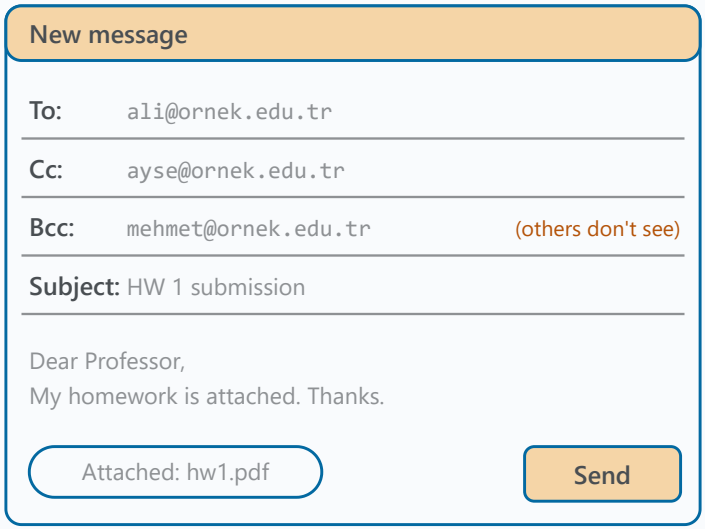
Separates the user from the domain.

Domain

Shows the organization that runs the mail service.

DIGITAL SERVICES

Parts of an Email



To

The main recipient.

Cc

People who receive the message for information.

Bcc

Recipients the others can't see.

Subject

A short, descriptive title for the message.

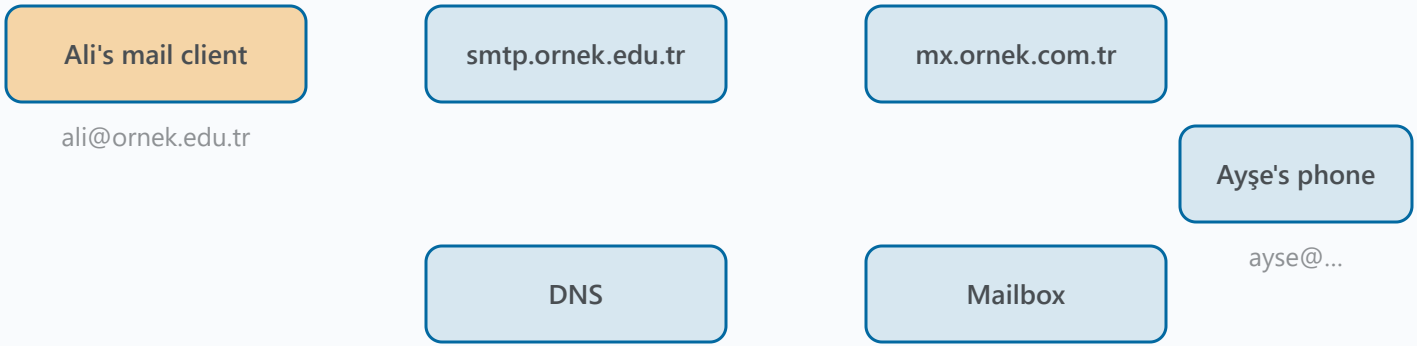
Attachment

A file sent with the message.

Reply / Forward

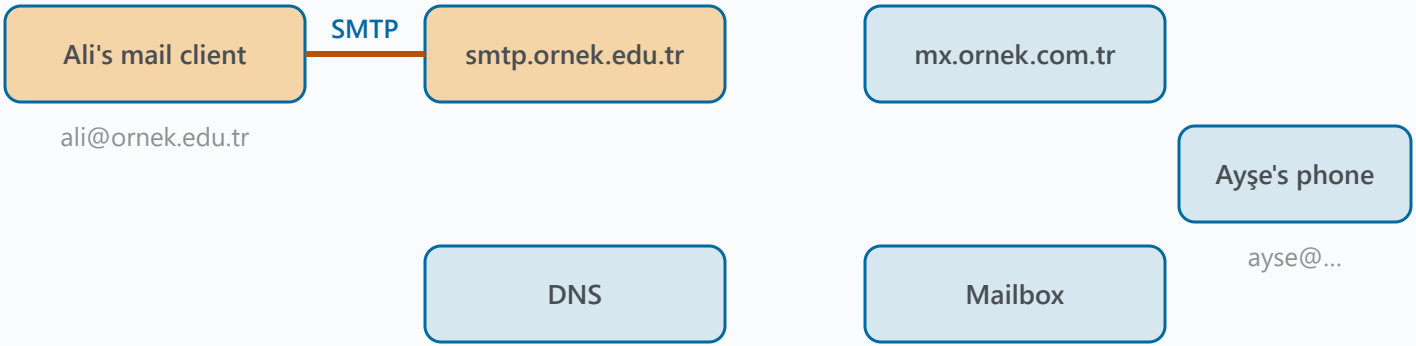
Answer the message or pass it on to someone else.

How Does an Email Get There?



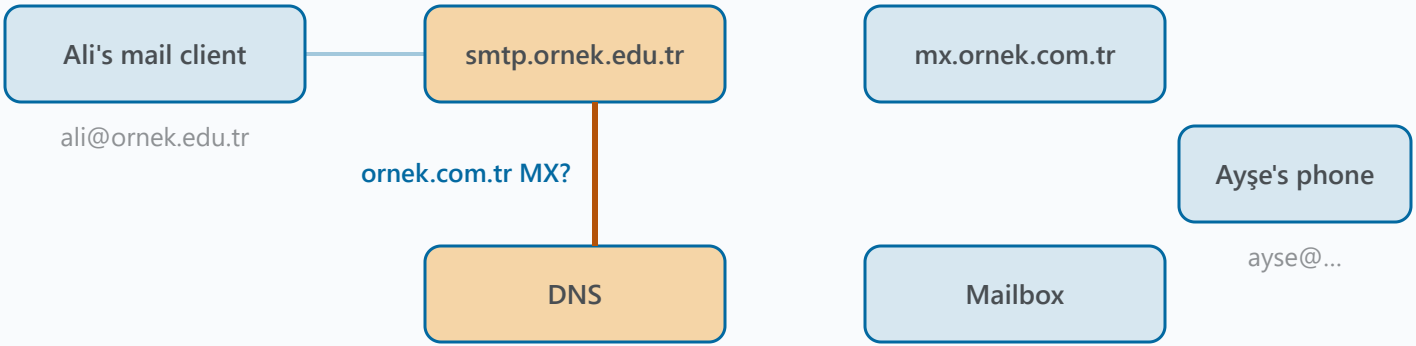
1. Ali writes a message to **ayse@ornek.com.tr** and clicks Send.

How Does an Email Get There?



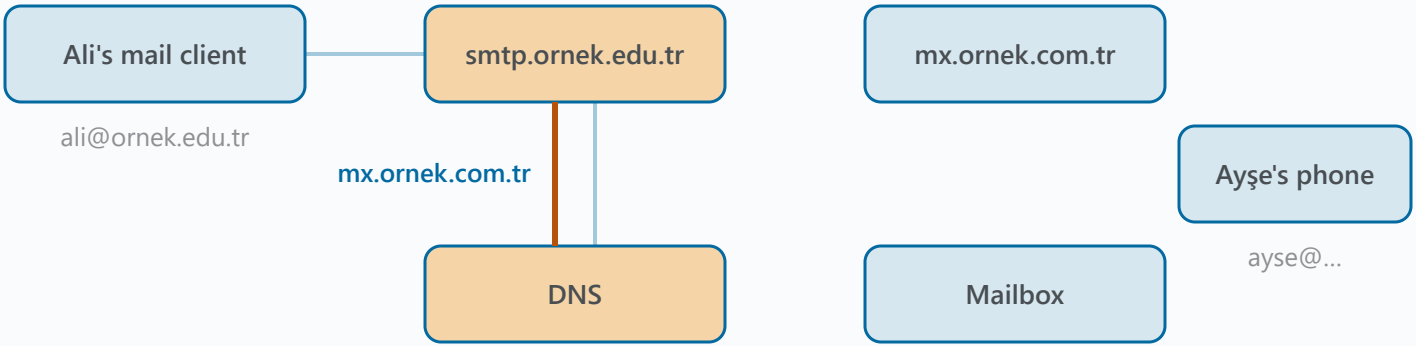
2. The client sends the message over **SMTP** to his organization's outgoing mail server.

How Does an Email Get There?



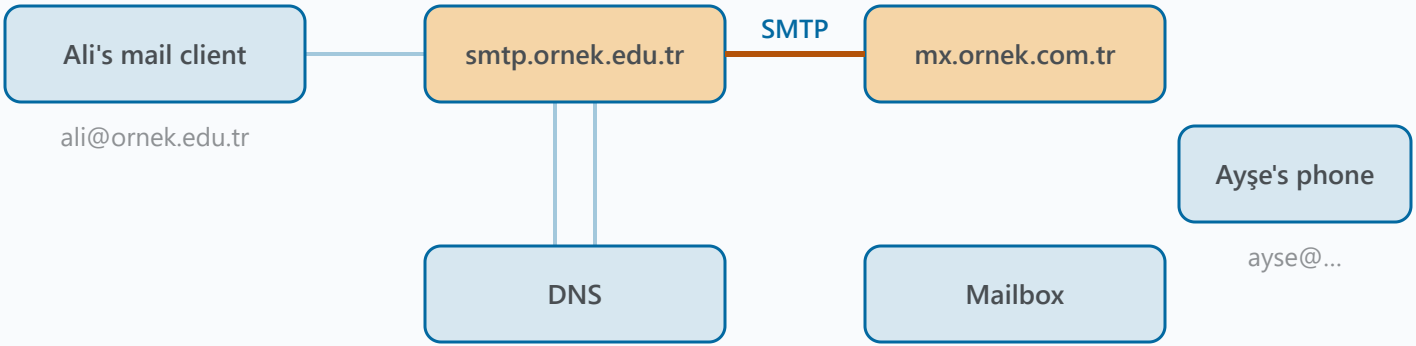
3. The server asks DNS for the mail server of the domain after the @ sign (the MX record).

How Does an Email Get There?



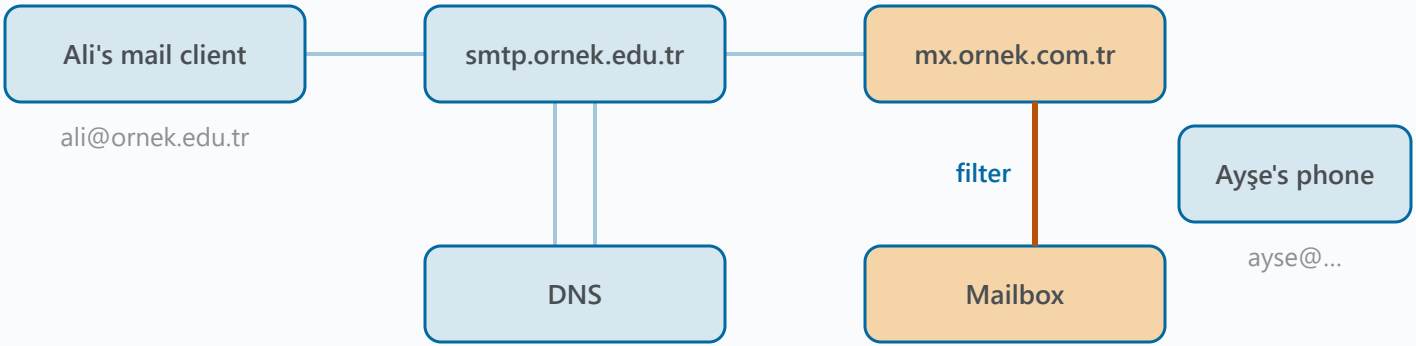
4. DNS answers: mail for ornek.com.tr is accepted by mx.ornek.com.tr.

How Does an Email Get There?



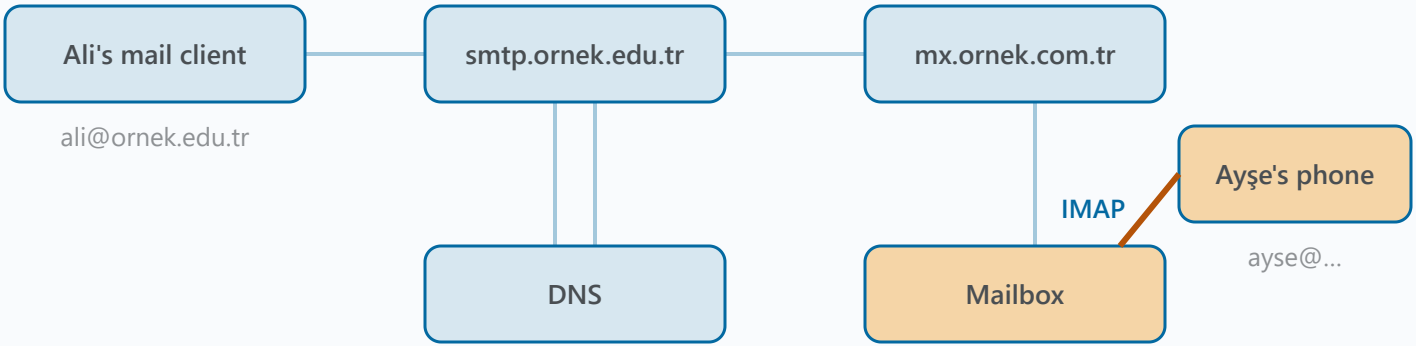
5. The message is transferred over SMTP to the recipient's mail server.

How Does an Email Get There?



6. The receiving server runs the message through spam/virus filters and stores it in Ayşe's mailbox.

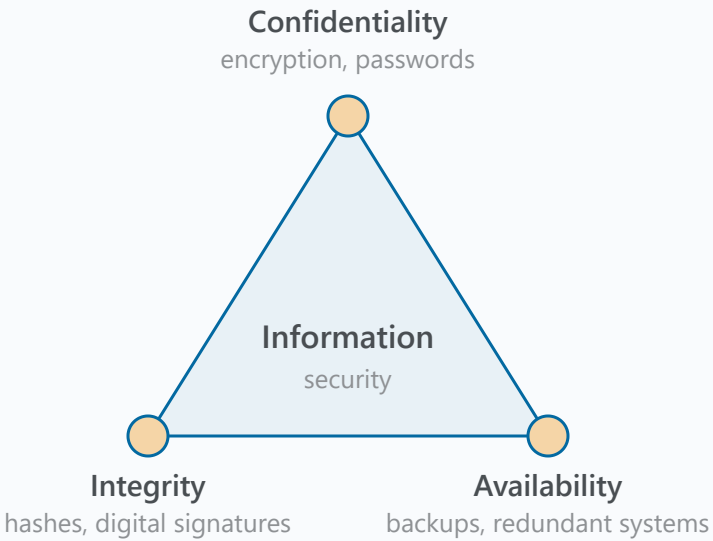
How Does an Email Get There?



7. Ayşe reads the message over **IMAP**. Since it stays on the server, it looks the same on her phone and computer.

SECURITY

The Three Goals of Information Security (CIA)



Confidentiality

Only authorized people have access. Threat: a stolen password.

Integrity

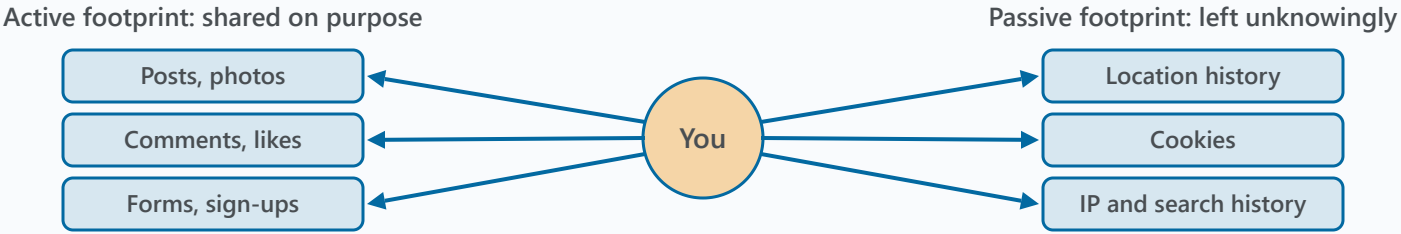
Information is not changed without permission. Threat: an altered grade record.

Availability

Information is there when needed. Threats: ransomware, a server crash.

SECURITY

Personal Data and Your Digital Footprint



Personal data

Information about an identified or identifiable person.

Digital footprint

The records left by your online activity.

Privacy

Controlling who your information is shared with, for what purpose and for how long.

Before sharing something, ask whether it is necessary, who will see it and whether it can be deleted later.

SECURITY

KVKK and e-Devlet Security

KVKK

Türkiye's Law No. 6698 on the **Protection of Personal Data** (KVKK) protects people's rights when their personal data is processed. Data controllers must state why data is processed and keep it secure.

e-Devlet

Türkiye's government services portal requires authentication. Choose a **strong password**, don't reuse it elsewhere, and turn on **two-step verification**. Never share your password.

For anything involving your credentials, type the address into the browser yourself instead of clicking links in emails and text messages.

SECURITY

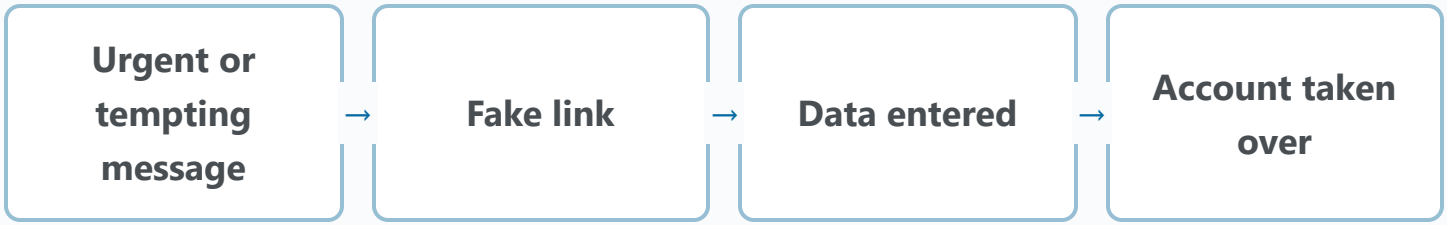
Malware

Type	Core behavior
Virus	Infects files and spreads when a user acts.
Worm	Can spread on its own over a network.
Trojan horse	Looks useful but hides a harmful function.
Ransomware	Makes files inaccessible and demands payment.
Spyware	Secretly monitors user activity.



SECURITY

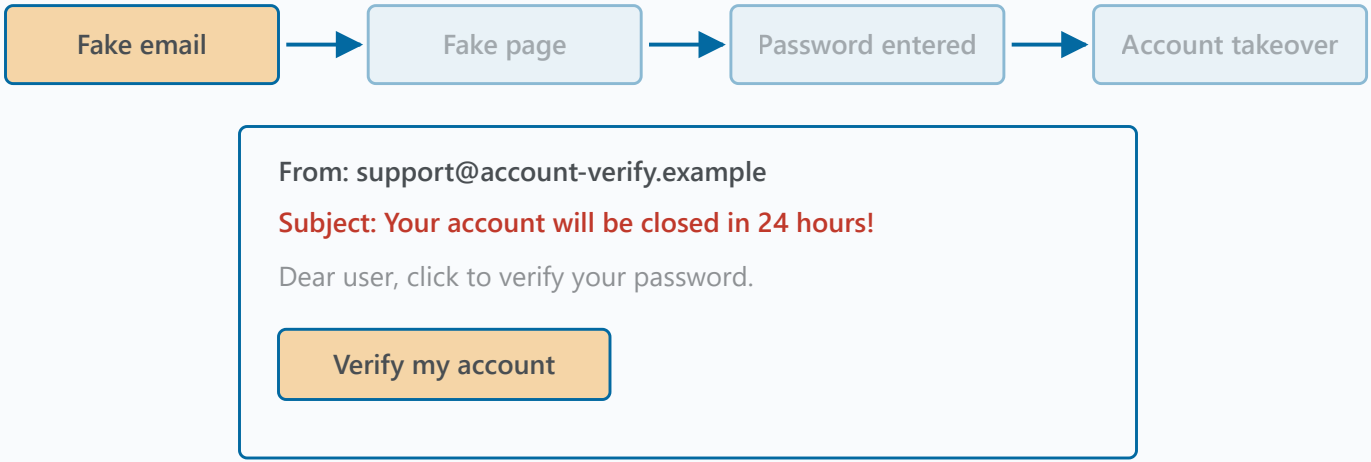
Social Engineering and Phishing



- Check the sender's address and domain carefully.
- Inspect the target address before opening a link.
- Never share passwords, card details or verification codes in a message.
- Verify a suspicious request through a separate channel.

SECURITY · STEP BY STEP

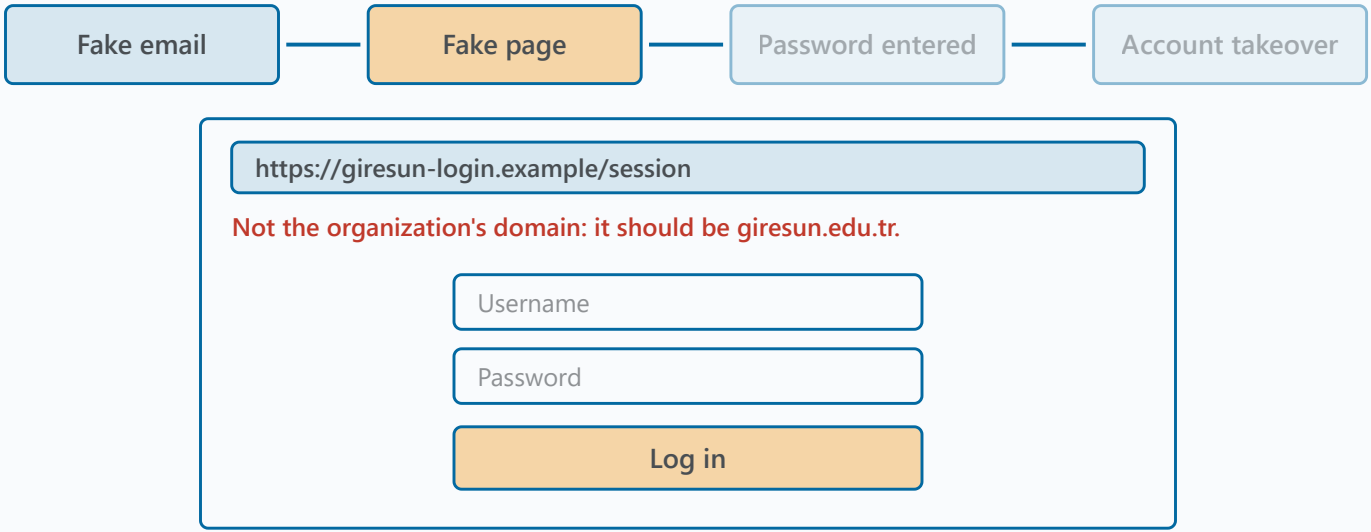
The Phishing Attack Chain and How to Break It



Goal: urgency and fear make you click without thinking.

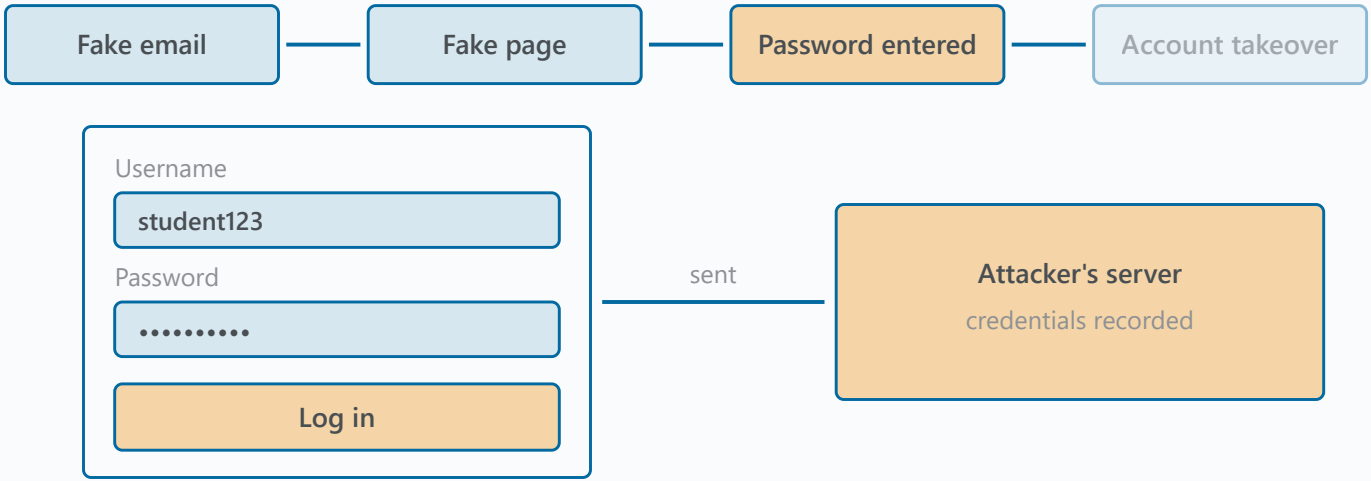
Fake email: the attacker sends an official-looking, urgent, threatening message. The sender's domain is not the organization's.

The Phishing Attack Chain and How to Break It



Fake page: the link leads to a login page that closely imitates the real one. The domain is different; https and the padlock only show that the connection is encrypted, not that the site is trustworthy.

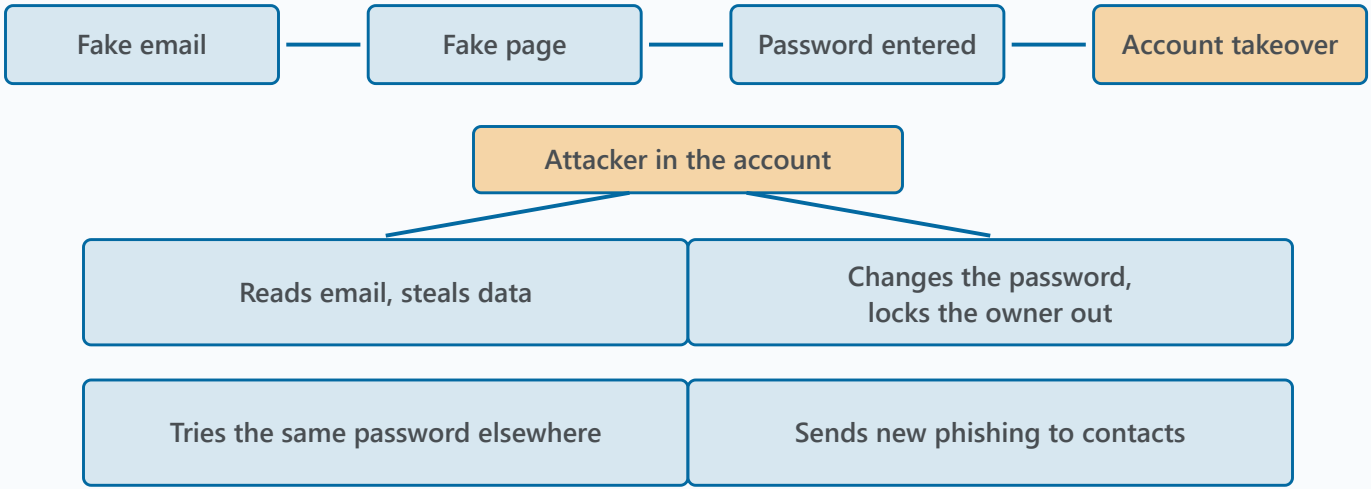
The Phishing Attack Chain and How to Break It



The victim is then usually sent to the real site.

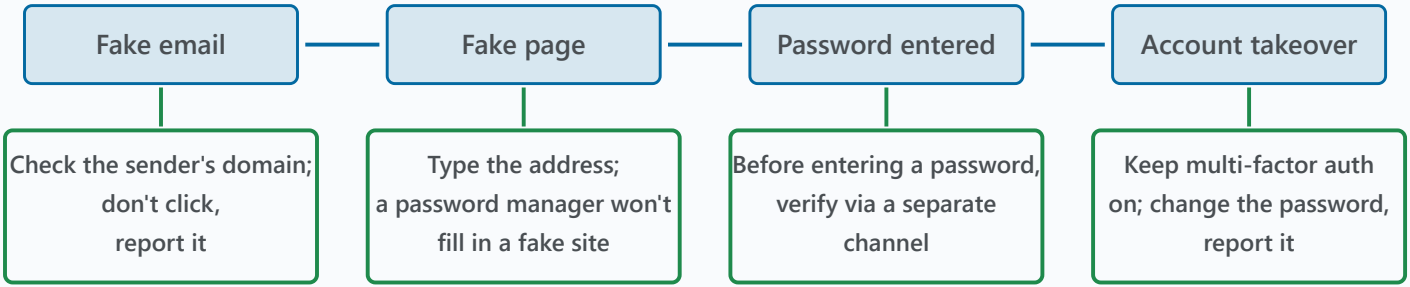
Password entered: the credentials the user types go straight to the **attacker's server**. The victim is often redirected to the real site and never notices.

The Phishing Attack Chain and How to Break It



Account takeover: the attacker logs in, may change the password to lock the owner out, tries the same password on other accounts and sends new phishing to the contacts.

The Phishing Attack Chain and How to Break It



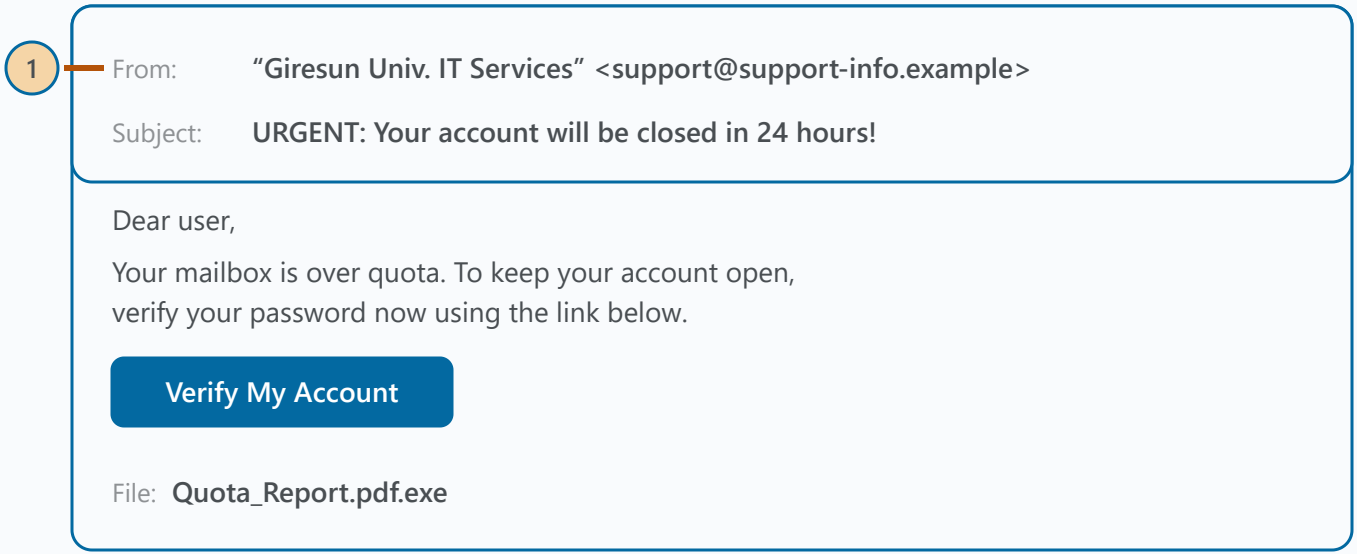
Every link is a breaking point:
break the chain anywhere and the attack fails.

When in doubt, go to the organization's known address, not the link.

Protection: every link in the chain is a chance to break it. Break one link and the attack fails; multi-factor authentication protects the account even if the password is stolen.

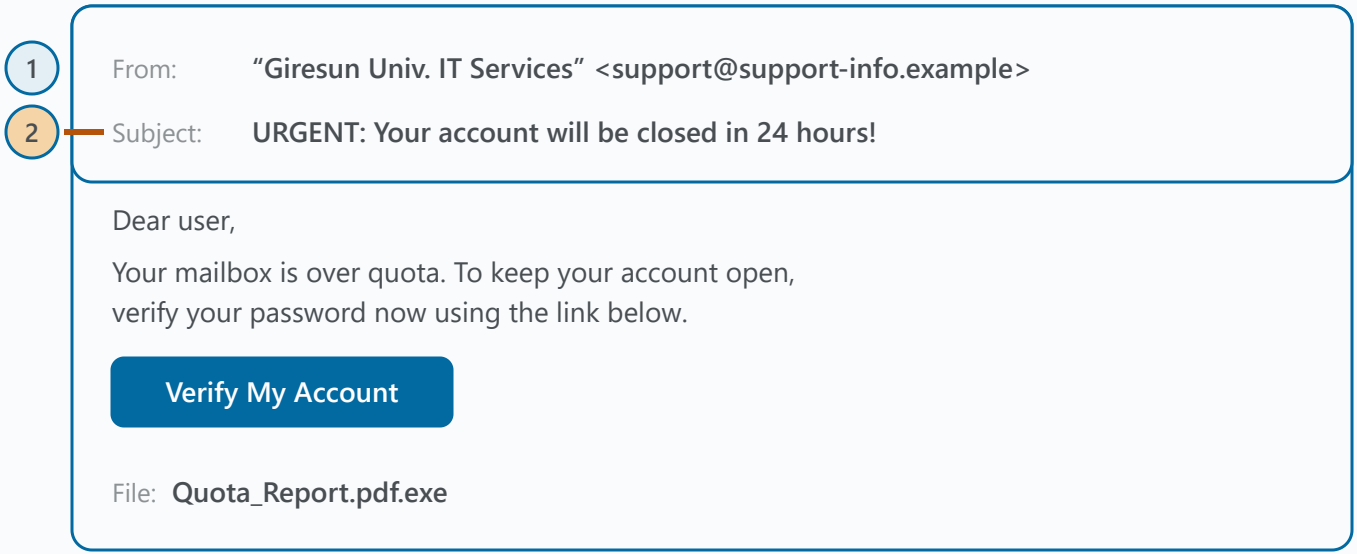
SECURITY

A Phishing Email, Annotated



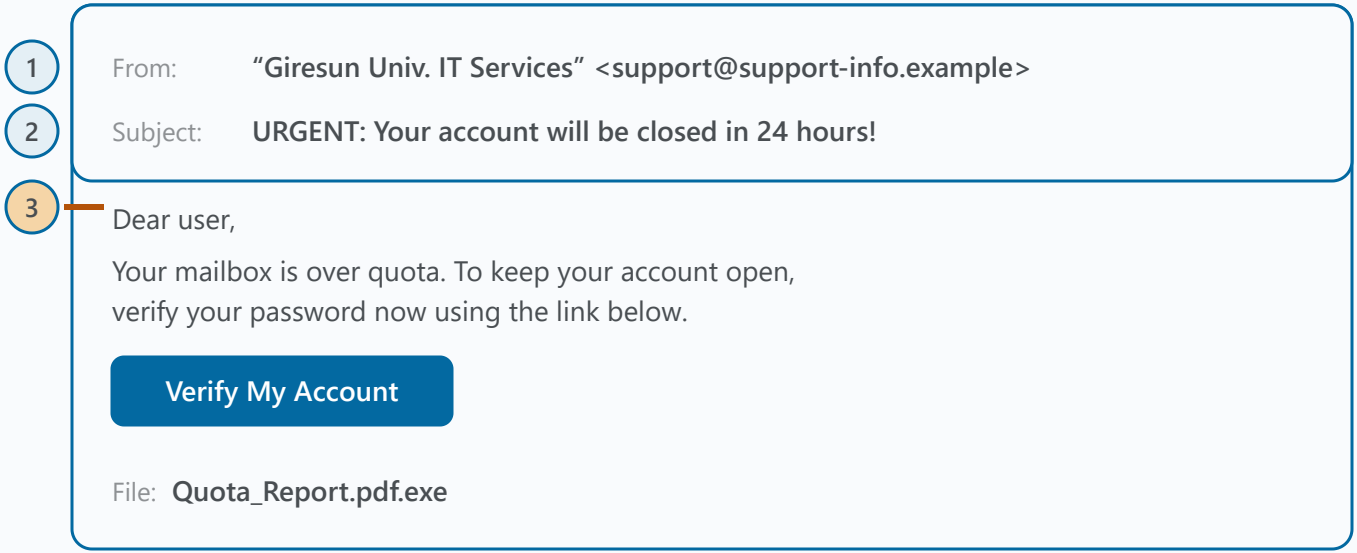
1. The sender: the display name looks official, but the domain is **support-info.example**; the real one should be giresun.edu.tr.

A Phishing Email, Annotated



2. The subject uses urgency and threats: “closed in 24 hours”. The goal is a click without thinking.

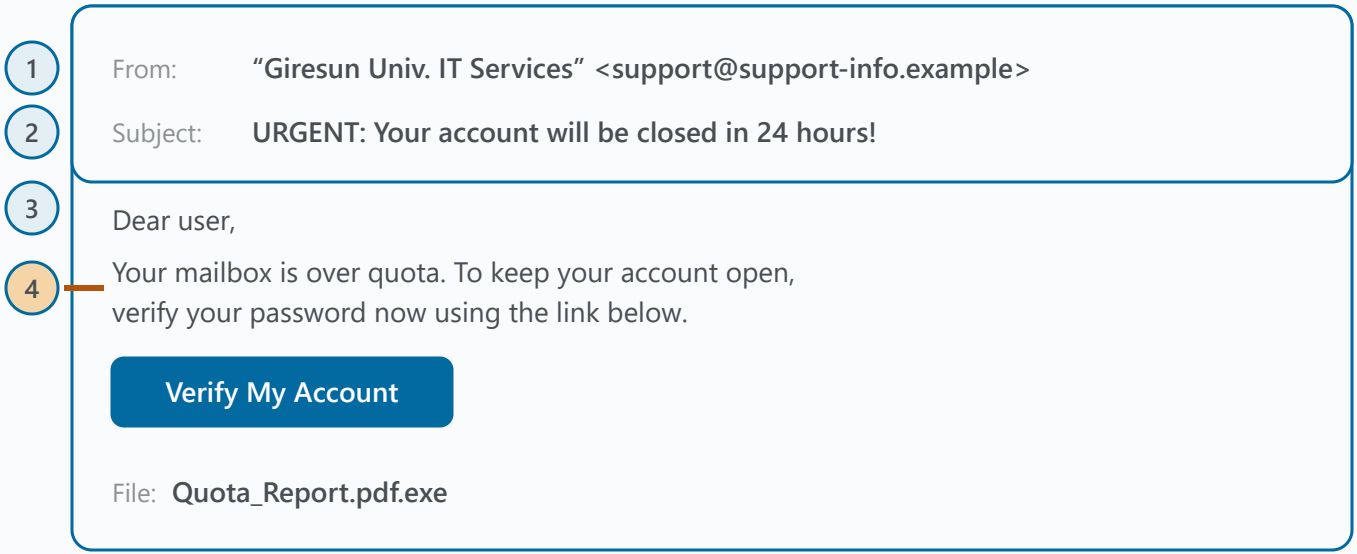
A Phishing Email, Annotated



3. A generic greeting (“Dear user”): your organization knows your name.

SECURITY

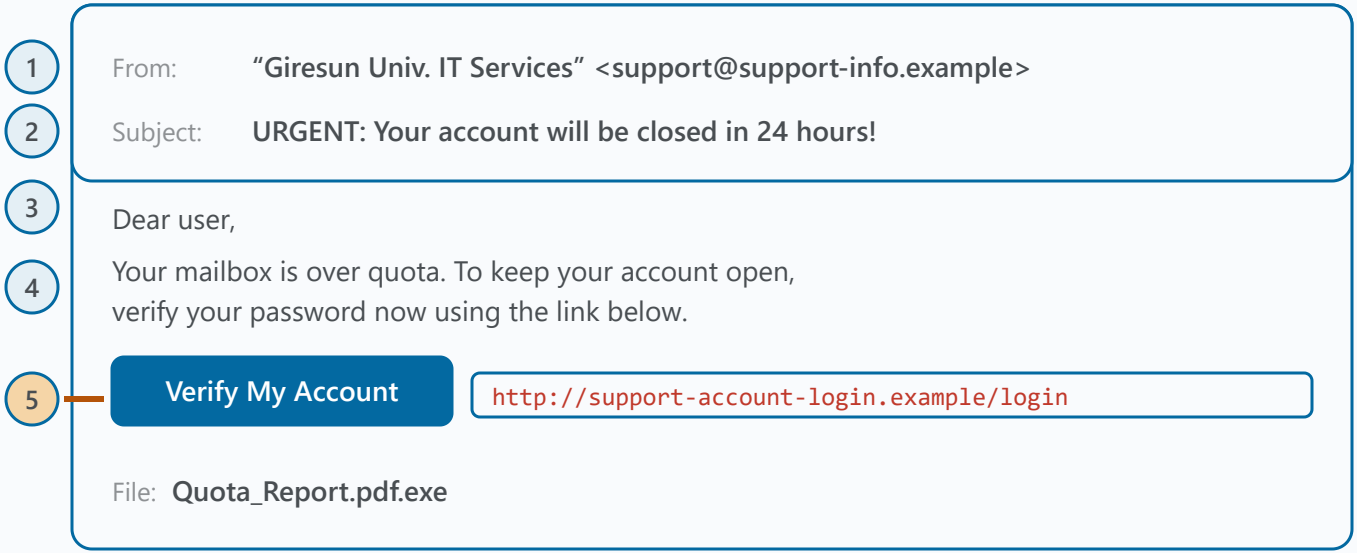
A Phishing Email, Annotated



4. A request to verify your password: real organizations never ask for it by email or through a link.

SECURITY

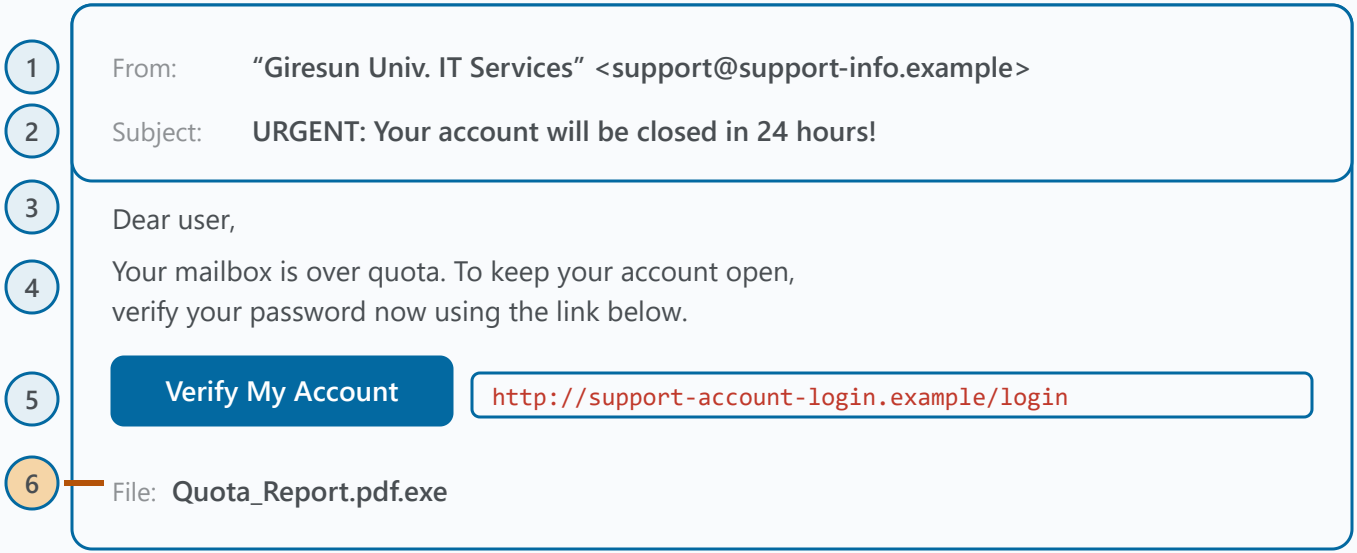
A Phishing Email, Annotated



5. Hovering over the button (without clicking) reveals that the real address points to a completely different domain.

SECURITY

A Phishing Email, Annotated



6. The attachment has a double extension: **.pdf.exe** is actually an executable program, not a document.

SECURITY

Safe Habits

Updates

Keep the operating system and applications up to date.

Passwords

Use a long, unique password for every account.

Multi-factor authentication

Adds a second layer of verification even if the password is stolen.

Backups

Back up important files to a separate, reliable location.

Downloads

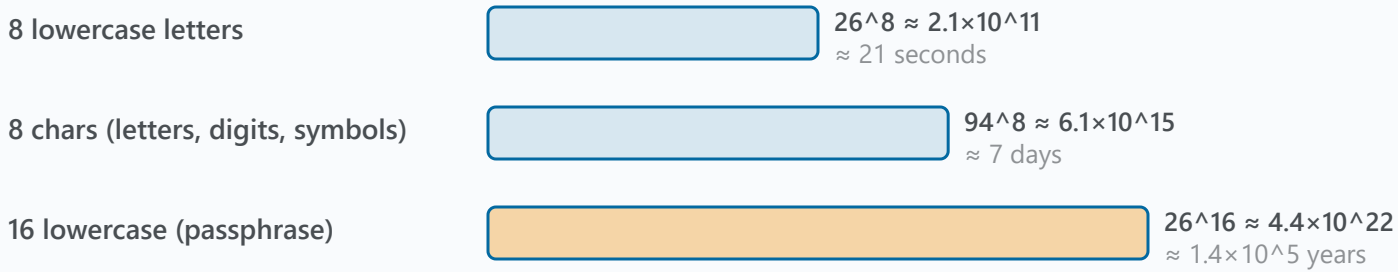
Get software only from trusted sources.

Connections

Be careful with sensitive tasks on shared networks.

SECURITY

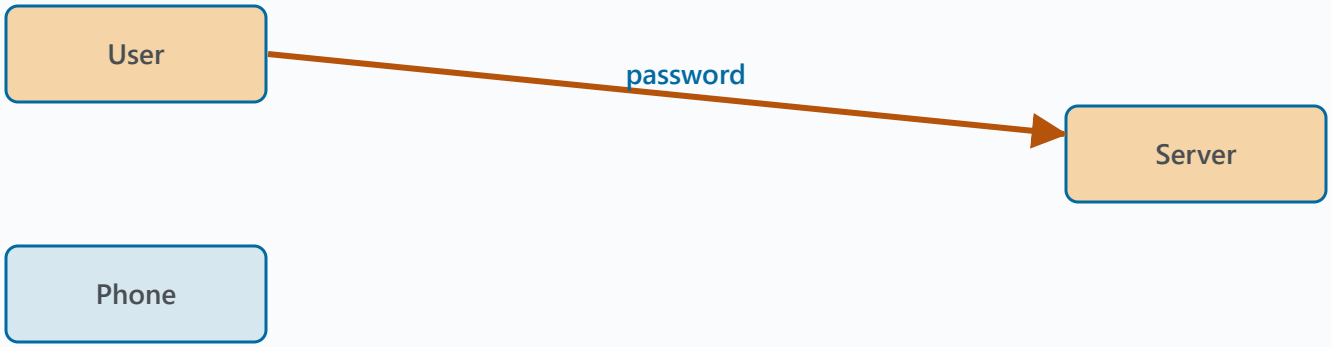
Password Strength: Why Length Matters



- Each added character multiplies the possibilities; length beats adding symbols.
- Passwords that are dictionary words, contain personal information or are reused elsewhere fall outside this calculation: they are easy to guess.
- A password manager generates and stores a long, unique password for every account.

SECURITY

Multi-Factor Authentication, Step by Step



1. The user enters a username and password (factor 1: something you **know**).

SECURITY

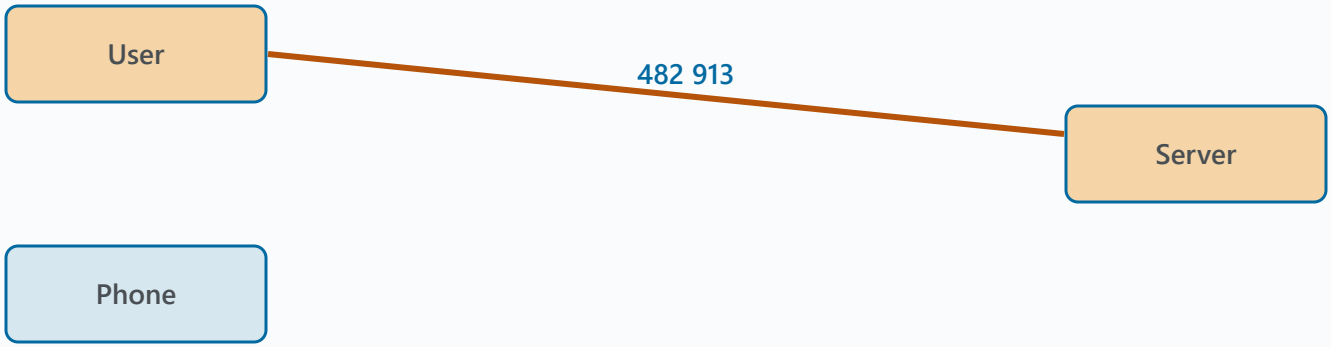
Multi-Factor Authentication, Step by Step



2. If the password is correct, the server asks for a second proof: an authenticator app on the phone generates a code that changes every 30 seconds (factor 2: something you **have**).

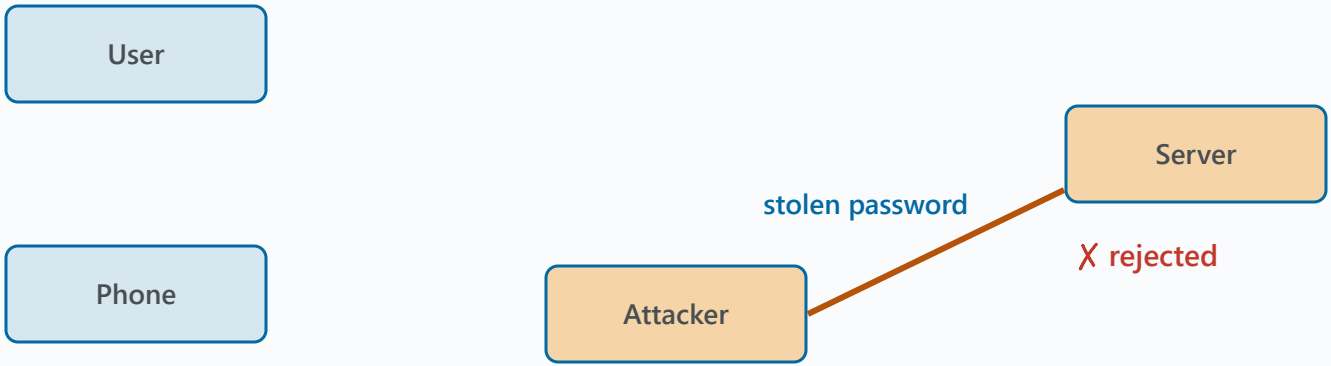
SECURITY

Multi-Factor Authentication, Step by Step



3. The user enters the code; the server verifies it and logs the user in.

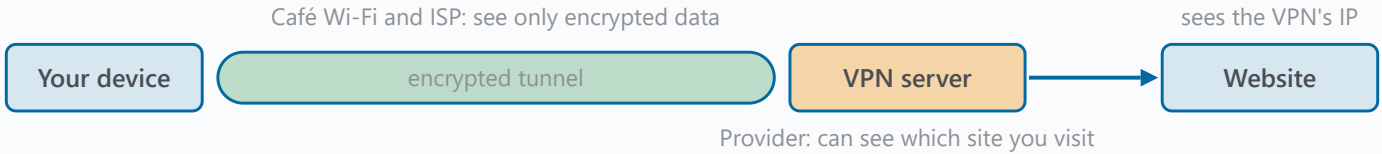
Multi-Factor Authentication, Step by Step



4. Even with a leaked password, the attacker usually **fails** without the phone. Real-time phishing and SIM swapping are still possible; the most resistant methods are FIDO2 passkeys and hardware keys.

SECURITY

VPNs: What They Do and Don't Do



What a VPN does

Builds an **encrypted tunnel** between your device and the VPN server. It makes local eavesdropping on public Wi-Fi harder, shows sites the VPN server's IP instead of yours, and gives secure remote access to a corporate network.

What a VPN doesn't do

It doesn't make you anonymous. The VPN provider can see which sites your traffic goes to, so you have to trust it. If you're logged in or have cookies, sites still recognize you. It doesn't block malware or phishing either.

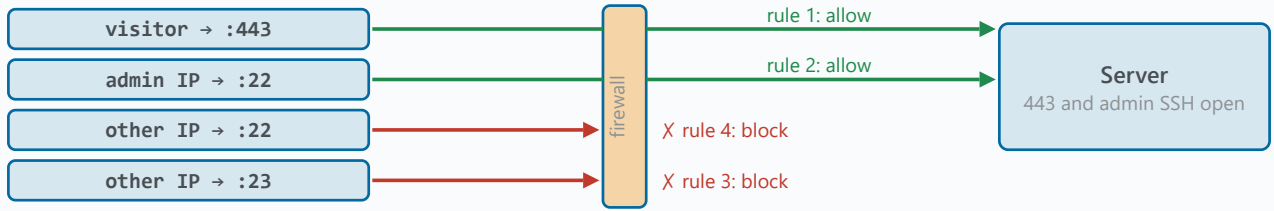
"No logs" claims are the provider's own statements. A VPN requires choosing a trustworthy provider; it complements HTTPS rather than replacing it.

SECURITY

Firewall Rules: An Example

A firewall allows or blocks incoming and outgoing connections **according to rules**; the rules are tried top to bottom and are usually based on IP and port.

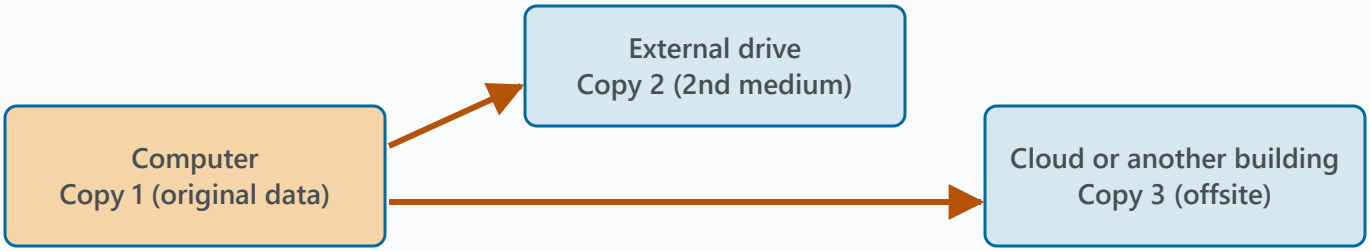
#	Direction	Dest. port	Source	Action
1	In	443 (HTTPS)	Anyone	**Allow** (web server)
2	In	22 (SSH)	Admin IP only	**Allow**
3	In	23 (Telnet)	Anyone	**Block** (unencrypted legacy protocol)
4	In	Everything else	Anyone	**Block** (default deny)



Good practice: block everything by default and allow only the ports you need. A firewall limits what gets into the network; on its own it won't stop a malicious file the user downloads.

SECURITY

Backups: The 3-2-1 Rule



3 copies · 2 different media · 1 copy offsite

3 copies

The original data and at least two backups.

2 media

Different kinds of storage, e.g. SSD and external drive, so they don't fail together.

1 offsite

In a different location (cloud or another building) to survive fire, theft or ransomware.

A backup is only useful if it can be restored: test it now and then. A drive left permanently connected can be hit by ransomware.

SUMMARY

Chapter Summary

- The Internet is a global infrastructure of networks connected by common protocols.
- An IP address identifies a device, a domain name gives it a human-readable name, and a URL identifies a specific resource.
- The Web is one of the services that run on the Internet.
- Search techniques speed up finding information; evaluating sources improves accuracy.
- Updates, unique passwords, multi-factor authentication and backups are the basic security measures.

Review Questions

1. What is the difference between the Internet and the Web?
2. Without DNS, how would we reach a website?
3. How do the jobs of a router and a switch differ?
4. Show the protocol, domain name and path of a URL.
5. What signs help you spot a phishing message?
6. Which three measures would you combine for strong account security?

SUMMARY

Exercises

1. At best, how many seconds does a 1.5 GB file take to download on a 200 Mbit/s line?
2. Split `https://obs.ornek.edu.tr/grades?term=2#midterm` into protocol, domain name, path, query and fragment.
3. Write the last octet of 192.168.10.37 as an 8-bit binary number.
4. List the DNS steps for a domain name that is not cached.
5. Your IP is 192.168.1.5, so why does a website see a different IP?
6. What three things would you check to tell whether a “parcel could not be delivered” text message is phishing?

▶ **Answers**

References

- Internet Society, A Brief History of the Internet.
- IETF, Internet Standards and RFC documents.
- W3C, Web Standards.
- NIST, Cybersecurity and Privacy guidance.

The End

Chapter 1.4: The Internet

Previous: Chapter 1.3: Computers · Next: Chapter 1.5: Computation and Computational Thinking