2024

How the web works

Inhaltsverzeichnis

Introduction to the Web **1**
Network Hardware *1*
Client-Server Architecture *1*
What is a Client? 1
What is a Server? 1
Operation Levels 2
Architecture Types 2
URL Format *3*
Domain Name System (DNS) *3*
Domain Space 4
DNS Response Types 4
DNS Server Modes 5
Why Do We Need Two Server Types? 5

Introduction to the Web

In this lesson, you will learn how data is exchanged between your browser and Internet server when you load a site, and what a link format and DNS are.

Network Hardware

Network hardware includes the physical devices used to build and maintain computer networks. The most common types are:

- **Routers** connect multiple networks and direct traffic between them.
- **Switches** connect multiple devices on a single network, enabling communication between them.
- **Hubs** are similar to switches but less intelligent, broadcasting data to all devices on the network.

Client-Server Architecture

What is a Client?

"Client" has two meanings. **First, it's an app** that connects to a server that determines the request type, receives the server's response, and confirms the process completion, like a web browser. **Second, it's the device running the app,** such as a phone, tablet, laptop, or desktop.
A client can display data received from the server and send user registration information from websites. Its logic is implemented using programming languages like Ruby, Java, PHP, Python, and JavaScript, and frameworks such as Ruby on Rails, Django, Laravel, and Flask.

What is a Server?

Server satisfies the client's information needs, functioning either as a database server or an application server. **Database server** provides data storage through relational or object-oriented DBMS like PostgreSQL, MySQL, or Oracle, while the **application server** executes programs and scripts on which applications are built.
There are also **web servers,** like Apache and Nginx, which handle about 50% of all requests. They forward these requests to application servers for processing, which often involves interaction with a database server. The application server then sends the processed response back to the web server via HTTP or HTTPS, which the web server delivers to the client.
In simplest terms:

1. **The web server** receives requests from the Internet and forwards them to the application server for processing.
2. **The application server** processes these requests and returns the results to the web server, such as registering a user or fetching a list of products from the database.
3. **The web server** sends the processed response back to the client.

Operation Levels

Logically, there are three levels of operations on which the client-server architecture works:

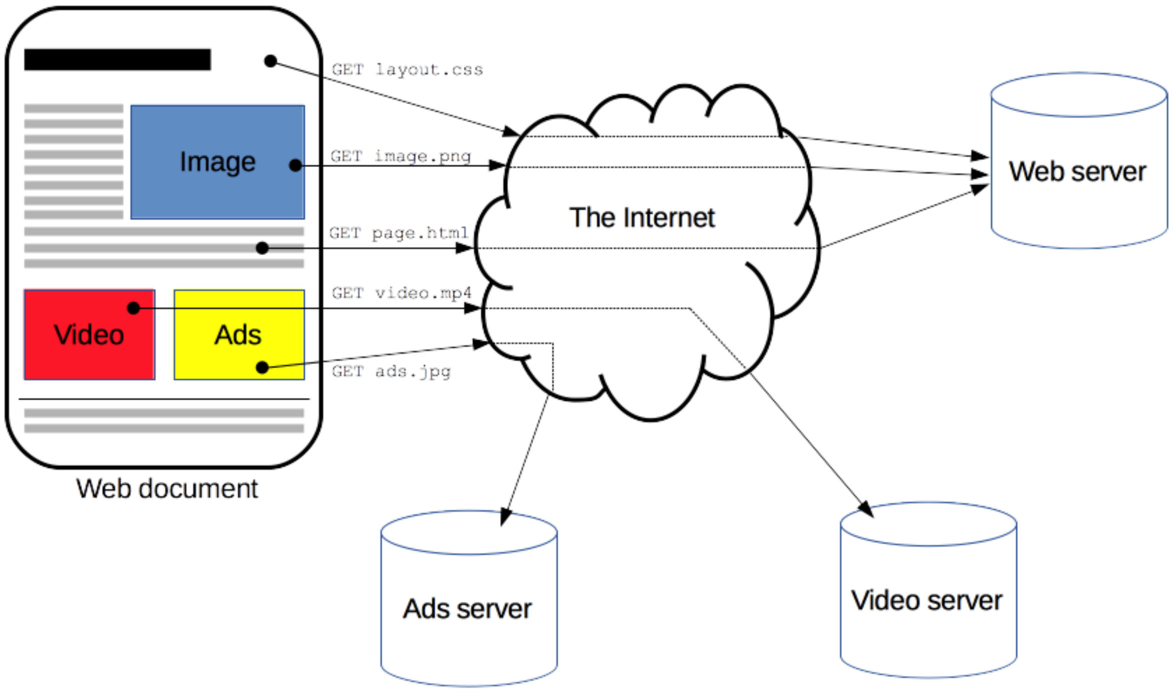
- **data presentation** — is responsible for user data displaying and entering control commands, for example, submitting a form for user registration;
- **application** — implements the main logic of the application; carries out the necessary information processing;
- **data management** — provides data storage in the database and its access.

Architecture Types

In a three-level client-server architecture, logic processing and user data storage are separated on different application servers, connected via an API. The client program calls server functions, known as "services," while application programs use SQL queries to access the database server. There are client types:

- **Thin client,** where all data operation and management logic is on the server; the client program only displays data to the user. One such example is a computer with a browser for web apps.
- **Thick client,** which processes data sent by the server using its own resources. Example: mobile applications that work offline using phone resources.

Below picture shows how our application interacts with different types of servers using API requests over the Internet.



URL Format

URL stands for **uniform resource locator** on the Internet, which we know as a link. Let's see what it can consist of, using `https://wikipedia.org/wiki/Computer` as an example:

- `http://` — protocol name, for example:
- `http` — data transfer
- `https` — encrypted data transfer
- `ftp` — file transfer
- `smtp` — simple mail transfer
- `wikipedia.org` — server domain name or IP address
- `/wiki/Computer` — address of a specific page to download

Domain Name System (DNS)

Domain Name System is a TCP/IP application layer protocol that converts IP addresses into human-readable text addresses and vice versa. This mechanism lets us change IPs while keeping the domain name intact, or assign multiple IPs to a single domain. As for finding a domain's IP, we simply enter `nslookup` in the command line (host and dig on Linux), e.g., `nslookup wikipedia.ord`.
Domains are structured like a tree:

1. Root domain (denoted by a dot, can be omitted).
2. High-level (first) domain:

- organization domains (usually in the US):
- `org` — non-commercial organizations.
- `com` — commercial organizations.
- `net`, `io` — computer networks organizations.
- country domains (`ua`, `uk`, `de`).

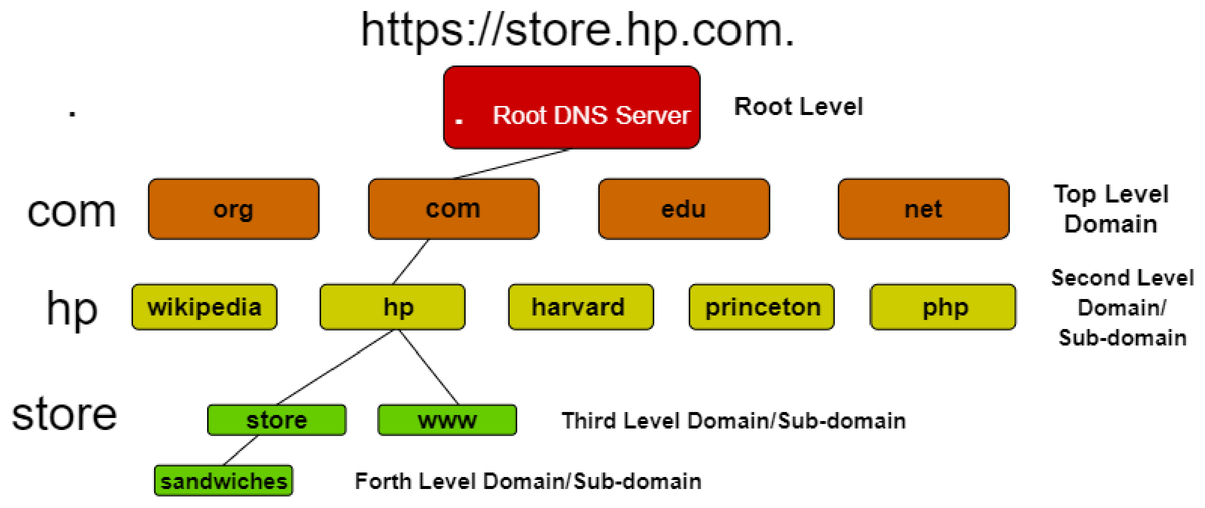
- domains in other languages (中国 is one of China's domains).

3. Second-level domains (`google`, `yahoo`).
4. Subdomains or computer addresses in the second-level domain (`www`, `maps`, `taxi`).

Domain Space

Domain space is a list of all computers and subdomain addresses within a specific domain. This space can include:

- **Root** contains records of all first-level subdomains.
- **Net** contains records of all second-level subdomains.
- **Pl** contains records of all subdomains and computers in the `pl` domain.

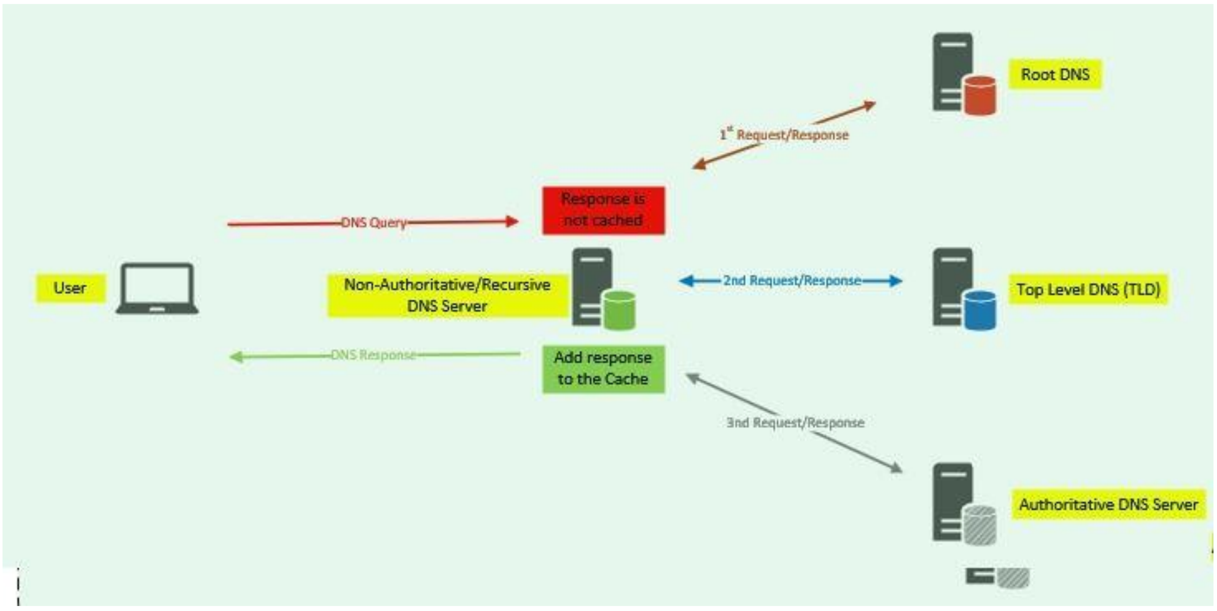


DNS Response Types

When a browser sends a request to a DNS server, the server searches for the corresponding IP address and returns it once found. There are two DNS response types:

- Authoritative: Provided by a server responsible for that domain space (iterative server).
- Non-authoritative: Provided by a server not responsible for that domain space, using cached responses from an authoritative server.

A DNS resolver is a computer that stores cached IP addresses, which first checks its cache for the required IP address and — if found — responds to the user. Otherwise, the DNS resolver sends requests to the authoritative servers. If the address isn’t found within a certain period, the DNS resolver returns an error.



DNS Server Modes

DNS server modes correspond to response types:

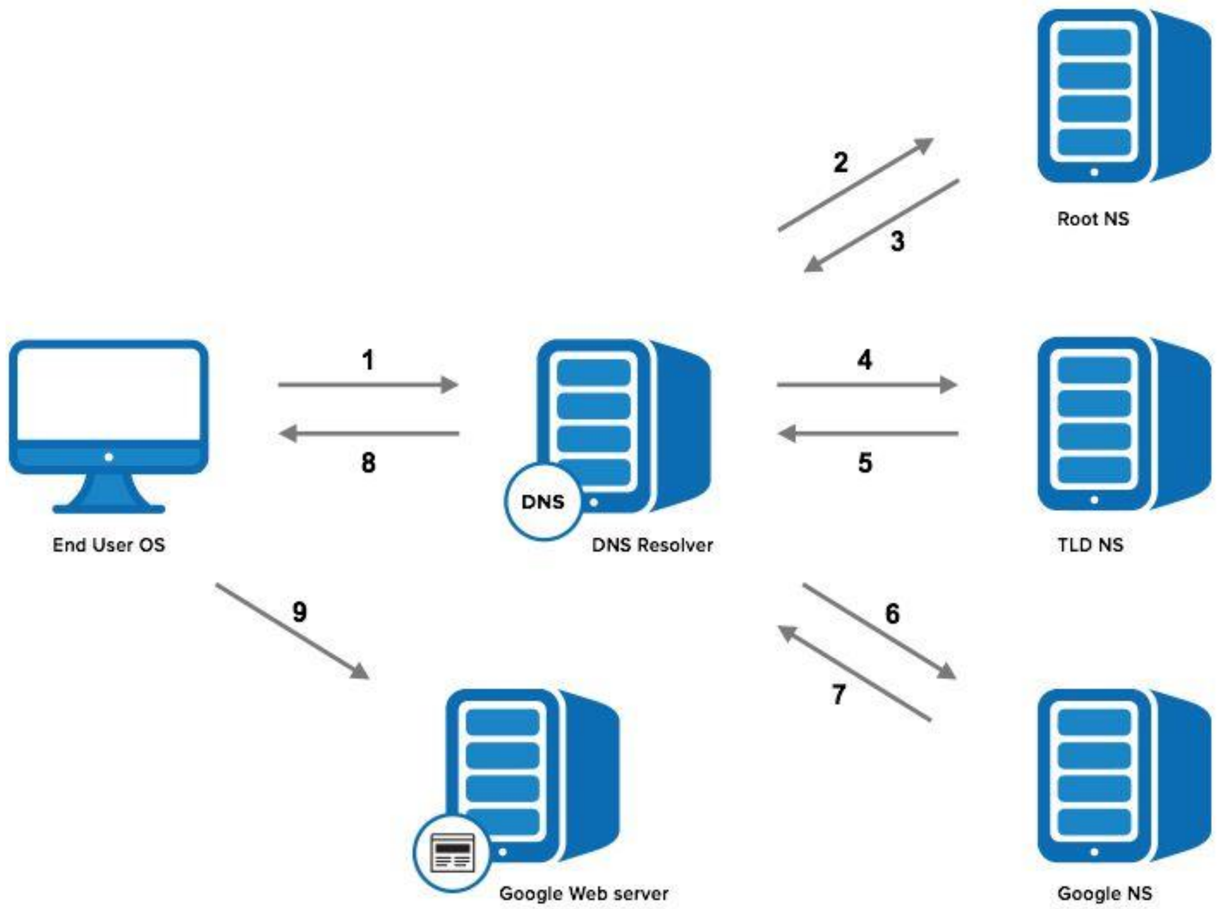
- **Authoritative,** in which the DNS server responds only if it is responsible for that domain space. Otherwise, it provides the address of another server to contact.
- **Recursive,** in which the DNS resolver queries multiple DNS servers until it finds the one with the desired IP address, then sends the received response to the client.

Why Do We Need Two Server Types?

DNS servers that store IP address information operate in authoritative mode. They handle numerous requests, especially for root and high-level domains.

Some servers, known as DNS resolvers, operate in recursive mode. They receive client requests, search the DNS server tree, obtain the response, and return it to the client. DNS resolvers are typically located on local networks, with their addresses often assigned automatically via DHCP along with the computer's IP address.

After a DNS resolver obtains an IP address for a domain name, it caches the information to improve efficiency. However, since IP addresses can change (albeit rarely), the cached response is marked as not authoritative and can be updated after a few days.



The DNS protocol operates on a "client-server" model. Clients can be either DNS clients or DNS servers running in recursive mode. Interaction occurs in a "request-response" manner without establishing a connection, using the UDP protocol on `port 53`.