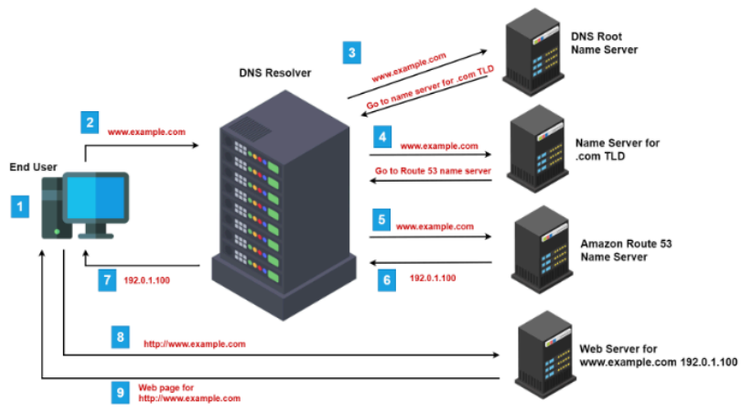
Domain Name System

Domain Name System (DNS) is a fundamental technology that plays a crucial role in how we access websites and services on the internet. It acts as a sort of “phonebook” for the internet, translating human-friendly domain names like “www.example.com” into the numerical IP addresses that computers and servers use to identify each other on the internet. This article will comprehensively discuss DNS service, its components, how it works, and its importance in the context of the internet.

Introduction to DNS

DNS, which stands for Domain Name System, is a distributed naming system that allows us to assign user-friendly domain names to the numeric IP addresses associated with websites, servers, and other network resources. Without DNS, we would need to remember complex IP addresses like “192.168.1.1” instead of simple domain names like “www.google.com.”

DNS is an essential part of the internet’s infrastructure and is often referred to as the “internet’s address book” because it provides the means to map human-readable domain names to the IP addresses that machines use to communicate with each other.



Components of DNS

The DNS system consists of several key components, each serving a specific role in the process of resolving domain names to IP addresses:

DNS Servers

DNS servers are specialized computers that store databases of domain names and their corresponding IP addresses. There are different types of DNS servers, including:

- **Root Servers:** These servers are at the top of the DNS hierarchy and store information about the top-level domains (TLDs) like .com, .org, and .net.
- **Top-Level Domain (TLD) Servers:** These servers manage domain names within specific TLDs (e.g., .com, .org, .gov).
- **Authoritative Name Servers:** These servers store DNS records for specific domains. Each domain typically has one or more authoritative name servers.
- **Recursive DNS Servers:** Also known as resolver servers, these servers interact with clients to resolve domain names by recursively querying other DNS servers until they find the authoritative server for a given domain.

DNS Records

DNS records are entries within a DNS database that provide information about a domain or subdomain. Common DNS record types include:

- **A Record:** Maps a domain name to an IPv4 address.
- **AAAA Record:** Maps a domain name to an IPv6 address.
- **CNAME Record:** Alias record that maps one domain name to another.
- **MX Record:** Specifies the mail servers responsible for receiving email on behalf of a domain.
- **TXT Record:** Allows domain owners to add arbitrary text information to a domain’s DNS record.
- **NS Record:** Specifies the authoritative name servers for a domain.

How DNS Works

Think of **DNS** as the **internet's phone book** .

You type a website name like www.example.com, but computers don't understand names — they only understand **numbers (IP addresses)**. DNS helps change the name into a number.

Let's see how it happens

Step 1: Your Computer Checks Its Memory

Your computer first asks:

> "Do I already know this website's number?"

- If it remembers, great!
- If not, it asks for help.

Step 2: Ask the Helper (DNS Server)

Your computer asks a **DNS helper server** (from your internet company or Google):

> "Do you know the number for this website?"

Step 3: Ask the Big Boss (Root Server)

If the helper doesn't know, it asks the **root server**:

> "Who knows about websites ending in **.com**?"

The root server replies:

> "Ask the **.com server**."

Step 4: Ask the .com Server

Now the helper asks the **.com server**:

> "Who knows about **example.com**?"

The .com server says:

> "Ask this server — it owns example.com."

Step 5: Ask the Website's Home Server

The helper asks the **authoritative server**:

> "What is the number for www.example.com?"

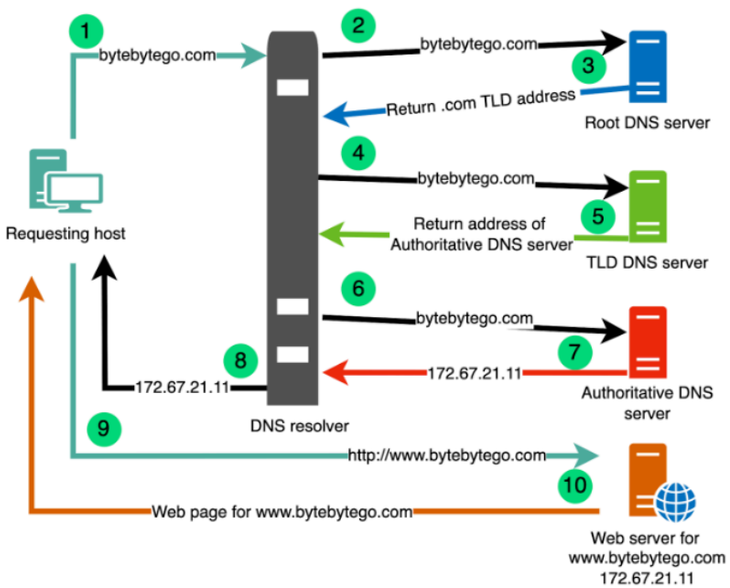
The server replies with the **IP address**

Step 6: Remember for Next Time

- The helper server saves the answer
- Your computer also saves it
  So next time, it's much faster

Step 7: Open the Website

Now your computer knows the number and goes directly to the website and shows you the page .



DNS Troubleshooting Commands – Detailed Explanation

When a website does not open, the problem usually falls into one of these areas:

- DNS resolution failure
- Network connectivity issue
- Server or application issue

These commands help identify **exactly where the problem is**.

---

1. `ping`

Command

```shell
ping example.com
```

What it actually does

1. First, your system tries to resolve the domain name using DNS.
2. If DNS succeeds, it sends small network packets (ICMP echo requests) to the IP address.
3. It waits for a response from the destination server.

Why it is important

- Confirms DNS resolution
- Confirms basic network connectivity

How to understand the output

- Replies with time values → DNS and network are working
- “Unknown host” → DNS failure
- No reply or packet loss → Network or firewall issue

When to use

- First step in any website troubleshooting
- Quick check to see if the issue is DNS or network-related

---

2. `nslookup`

Command

```shell
nslookup example.com
```

What it actually does

- Sends a DNS query to the configured DNS server
- Requests the IP address for the domain name
- Displays which DNS server provided the answer

Why it is important

- Confirms whether DNS servers can resolve the domain
- Helps identify DNS server problems

Output explanation

- `Server` → DNS server used
- `Address` → DNS server IP
- `Name` and `Address` → Resolved domain and IP

Common errors

- `NXDOMAIN` → Domain does not exist
- Timeout → DNS server unreachable

---

3. `dig`

Command

```shell
dig example.com
```

What it actually does

- Performs a full DNS query
- Shows all DNS response details
- Displays query time and response status

Why it is important

- Most reliable DNS troubleshooting tool
- Used by system administrators and DevOps engineers

Key sections explained

- `HEADER` → Query status (NOERROR means success)
- `QUESTION SECTION` → What was asked
- `ANSWER SECTION` → IP address or record
- `Query time` → DNS response speed

Common status values

- `NOERROR` → DNS working correctly
- `NXDOMAIN` → Domain does not exist
- `SERVFAIL` → DNS server error

---

4. Using a Public DNS Server

Command

```shell
dig example.com @8.8.8.8
```

What it actually does

- Sends the DNS query directly to Google's DNS server
- Skips your local or ISP DNS servers

Why it is important

- Helps determine if the issue is with your DNS provider
- Confirms whether the domain is publicly resolvable

How to interpret

- Works with public DNS but not local DNS → Local DNS issue
- Fails everywhere → Domain or authoritative DNS issue

---

5. `host`

Command

```shell
host example.com
```

What it actually does

- Performs a simple DNS lookup
- Returns IP address or mail server records

Why it is important

- Fast and clean output
- Useful for quick checks

When to use

- When detailed output is not required
- During quick validation tasks

---

6. `curl`

Command

```shell
curl example.com
```

What it actually does

1. Resolves DNS
2. Connects to the server using HTTP/HTTPS
3. Requests data from the server

Why it is important

- Tests DNS, network, and application layer together
- Helps detect server-side issues

Common outputs

- HTML or response text → Server working
- Connection refused → Server down or port closed
- Timeout → Network or firewall issue

---

7. `traceroute` / `tracert`

Command (Linux)

```shell
traceroute example.com
```

What it actually does

- Shows every network hop between your system and the destination
- Measures response time at each step

Why it is important

- Identifies where the connection breaks
- Helps isolate routing or ISP issues

How to read it

- Stops early → Network or firewall block
- Reaches destination → Network path is clear

---

8. Checking DNS Configuration

Command

```shell
cat /etc/resolv.conf
```

What it actually does

- Displays configured DNS servers
- Shows search domains

Why it is important

- Ensures the system is using correct DNS servers
- Helps identify misconfiguration

---

9. Clearing DNS Cache

Purpose

DNS records are cached to improve speed.
Sometimes cached data becomes outdated or incorrect.

Commands

Linux

```shell
sudo systemctl restart systemd-resolved
```

Why it is important

- Forces the system to request fresh DNS data

- Fixes issues caused by stale records

Recommended Troubleshooting Flow

```
1. ping domain
2. nslookup domain
3. dig domain
4. dig domain using public DNS
5. curl domain
6. traceroute domain
7. check DNS configuration
8. clear DNS cache
```

Final Summary

- `ping` verifies reachability
- `nslookup` checks basic DNS resolution
- `dig` provides full DNS analysis
- Public DNS tests isolate provider issues
- `curl` confirms server response
- `traceroute` identifies network path issues
- DNS cache clearing resolves stale data problems
