Dynamic Network Services
DHCP, DNS, and NAT

Prepared By
Dr. Islam Zakaria

Resources

Book: **Data.Communications.and.Networking.5th.Edition**

- **Chapter 18.4.4: Dynamic Host Configuration Protocol (DHCP)**
- **Chapter 18.4.5: Network Address Resolution (NAT)**
- **Chapter 26.6: Domain Name System (DNS)**

Useful Videos

DHCP + Wireshark

https://www.youtube.com/watch?v=_rfTm3hz3dE&list=PLgRZV-Axn6g7LUGzGPB0PxBOrck3viUoM&index=3

DNS + Wireshark

https://www.youtube.com/watch?v=jQBVzSuv5M4&list=PLgRZV-Axn6g7LUGzGPB0PxBOrck3viUoM&index=4

NAT + Wireshark

https://www.youtube.com/watch?v=rDAKKiVZPIs&list=PLgRZV-Axn6g7LUGzGPB0PxBOrck3viUoM&index=7

Lecture Objectives

By the end of this lecture, students should be able to:

- Explain the purpose and working mechanisms of DHCP, DNS, and NAT.
- Understand their roles in IP address allocation, name resolution, and address translation.
- Analyze message formats, communication sequences, and interactions among these protocols.
- Configure and troubleshoot DHCP, DNS, and NAT in simulated or real network environments.

Introduction: The Need for Dynamic Network Services

The Core Problem (In early IP networks):

- IP addresses and network parameters were manually configured.
- Hostnames were statically mapped in local files (/etc/hosts).
- Every device required a globally unique IP address to access the Internet.

As networks scaled to thousands of devices and mobile users, manual management became impractical and error-prone.

The Three Fundamental Services

Protocol	Main Function	Problem Solved
DHCP	Automatically assigns IP configuration	Eliminates manual IP configuration
DNS	Translates human-readable names (friendly address) to IP addresses	Eliminates the need to memorize IPs
NAT	Translates private IPs to public IPs	Solves IPv4 address shortage and provides isolation

DHCP

Dynamic Host Configuration Protocol (DHCP)

In early networks:

- Network administrators manually assigned IP addresses, subnet masks, and gateways.
- This caused **conflicts**, **duplication**, and **configuration errors**.
- Mobile devices (laptops, phones) couldn't easily change networks.
- **DHCP automates this by dynamically assigning IP parameters when a device joins the network.**

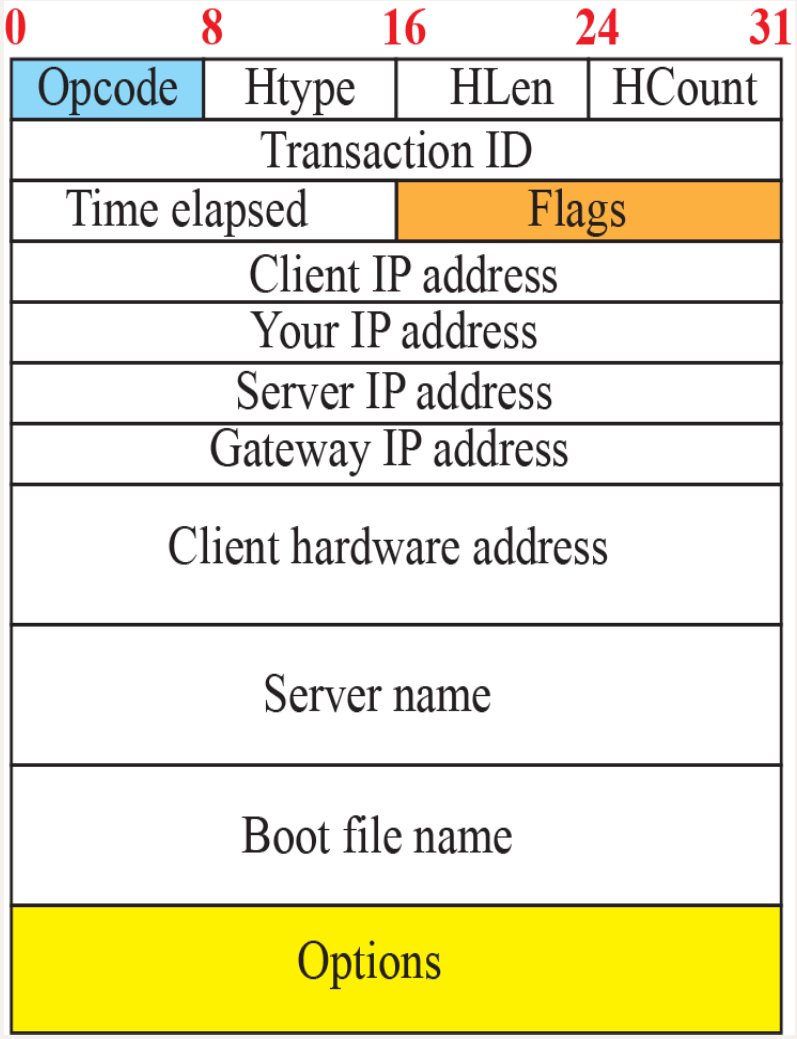
DHCP Overview

- ❑ Application-layer protocol (uses UDP ports **67**/**68**)
- ❑ Operates in a **client–server model**
- ❑ Based on **broadcast communication** within a local network

Can automatically provide:

- ➢ **IP address**
- ➢ **Subnet mask**
- ➢ **Default gateway**
- ➢ **DNS server address**
- ➢ **Lease duration**

DHCP Message Format



Fields:

Opcode: Operation code, request (1) or reply (2)
Htype: Hardware type (Ethernet, ...)
HLen: Lengh of hardware address
HCount: Maximum number of hops the packet can travel
Transaction ID: An integer set by client and repeated by the server
Time elapsed: The number of seconds since the client started to boot
Flags: First bit defines unicast (0) or multicast (1); other 15 bits not used
Client IP address: Set to 0 if the client does not know it
Your IP address: The client IP address sent by the server
Server IP address: A broadcast IP address if client does not know it
Gateway IP address: The address of default router
Server name: A 64-byte domain name of the server
Boot file name: A 128-byte file name holding extra information
Options: A 64-byte field with dual purpose described in text

DHCP Operation (The DORA Process)

DORA = Discover → Offer → Request → Acknowledge

1. **DHCPDISCOVER (Client → Broadcast):** “I need an IP address.”
   - The client, with an IP of **0.0.0.0**, **broadcasts** a "DHCP Discover" message.
   - Source Port: **68** (Client)
   - Destination Port: **67** (Server)

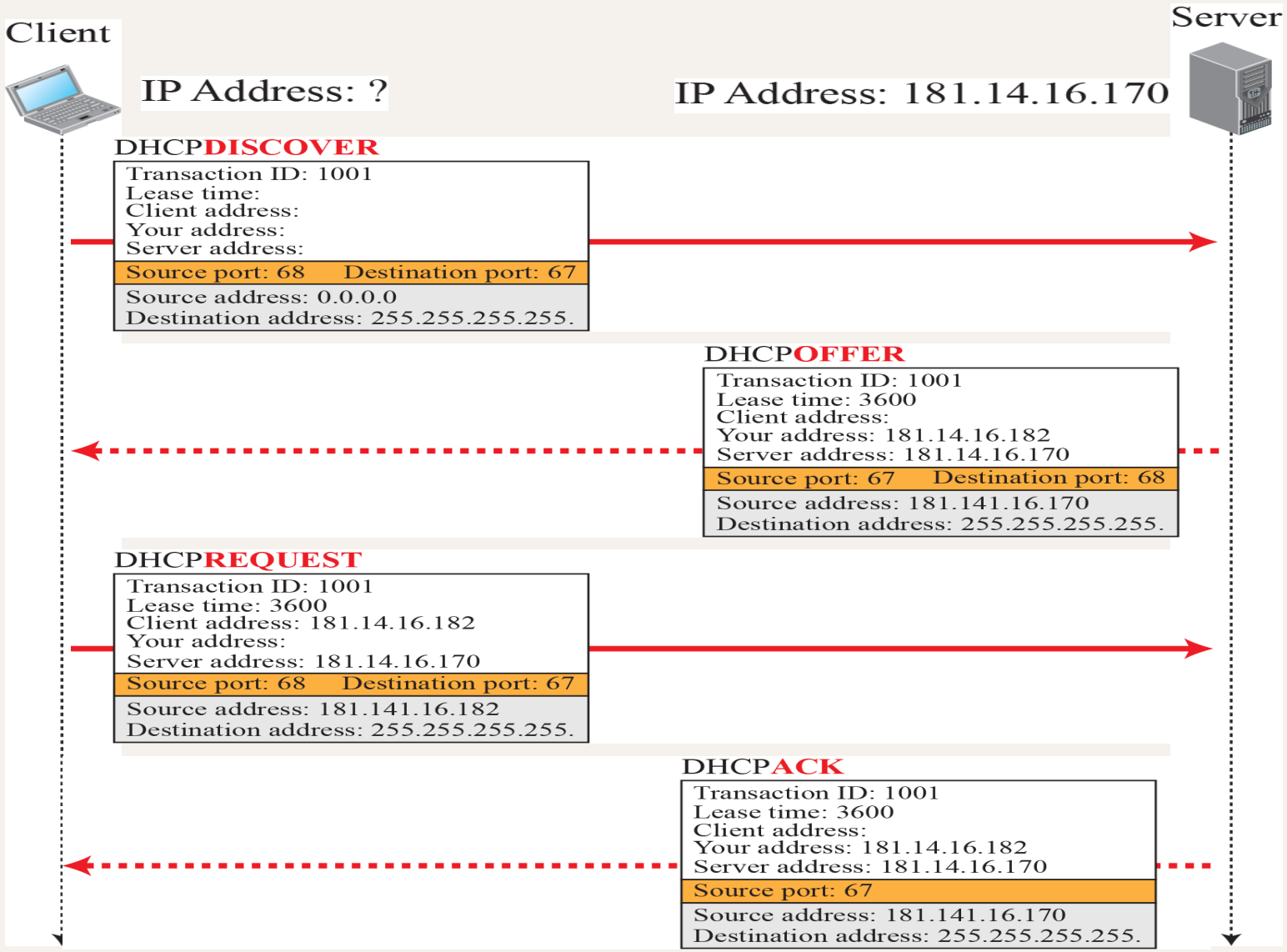
DHCP Operation (The DORA Process)

2. **DHCPOFFER (Server → Broadcast):** “I can offer you 192.168.1.10.”
   - The server responds with a "DHCP Offer" message, offering an IP address.
   - Source Port: **67** (Server)
   - Destination Port: **68** (Client)

DHCP Operation (The DORA Process)

3. **DHCPREQUEST (Client → Broadcast):** “I request 192.168.1.10 from that server.”
   - The client broadcasts a "DHCP Request" message, accepting the offered IP.
4. **DHCPACK (Server → Broadcast):** “Confirmed. You can use that IP for 8 hours.”
   - The server sends a final "DHCP ACK" message, confirming the lease.

DHCP Operation (Cont.)



Address Allocation Methods

1. **Dynamic allocation**: Temporary, with lease renewal.
   - IP addresses are temporarily assigned to devices from a pool, with a lease time after which the device must renew the lease or get a new address.
   - This allows flexible and efficient use of IP address space for transient or mobile devices.
2. **Automatic allocation**: Permanent assignment.
   - Similar to dynamic, but the client keeps the assigned IP address permanently unless manually released.
   - The server automatically assigns a permanent address from a pool without manual intervention.

Address Allocation Methods (Cont.)

3. **Manual allocation (reservation)**: IP fixed for specific MAC address.
   - Specific IP addresses are permanently reserved for particular devices based on their MAC addresses.
   - This ensures those devices always get the same IP, useful for servers or resources needing consistent addressing.

DHCP Lease Renewal

DHCP Lease Renewal is a process that occurs before the expiration of the current IP address lease to maintain continuous network connectivity for the client device.

The process follows these steps:

1. When the lease time reaches **50%**, the DHCP client sends a **DHCPREQUEST** message to the DHCP server that granted the lease, requesting renewal.
2. The DHCP server responds with a **DHCPACK** message, confirming and extending the lease.
3. If the server does not respond, the client enters the **rebinding phase** and **broadcasts a DHCPREQUEST to any available DHCP server**.
4. The client continues retrying until it receives a **DHCPACK** or the lease expires.
5. If the lease expires without renewal, the client must stop using the IP address and start the process to obtain a new lease from scratch.

DNS

Domain Name System (DNS)

The Problem DNS Solves

Originally, hostnames were manually stored in a central text file called **hosts.txt** (*C:\Windows\System32\drivers\etc\hosts*)

As the Internet grew:

- Managing and distributing this file became impossible.
- Users needed a scalable way to **map names to IP addresses automatically**.
- **DNS** provides a **distributed, hierarchical database** to resolve names efficiently.

DNS Overview

Application layer protocol using UDP/TCP port 53

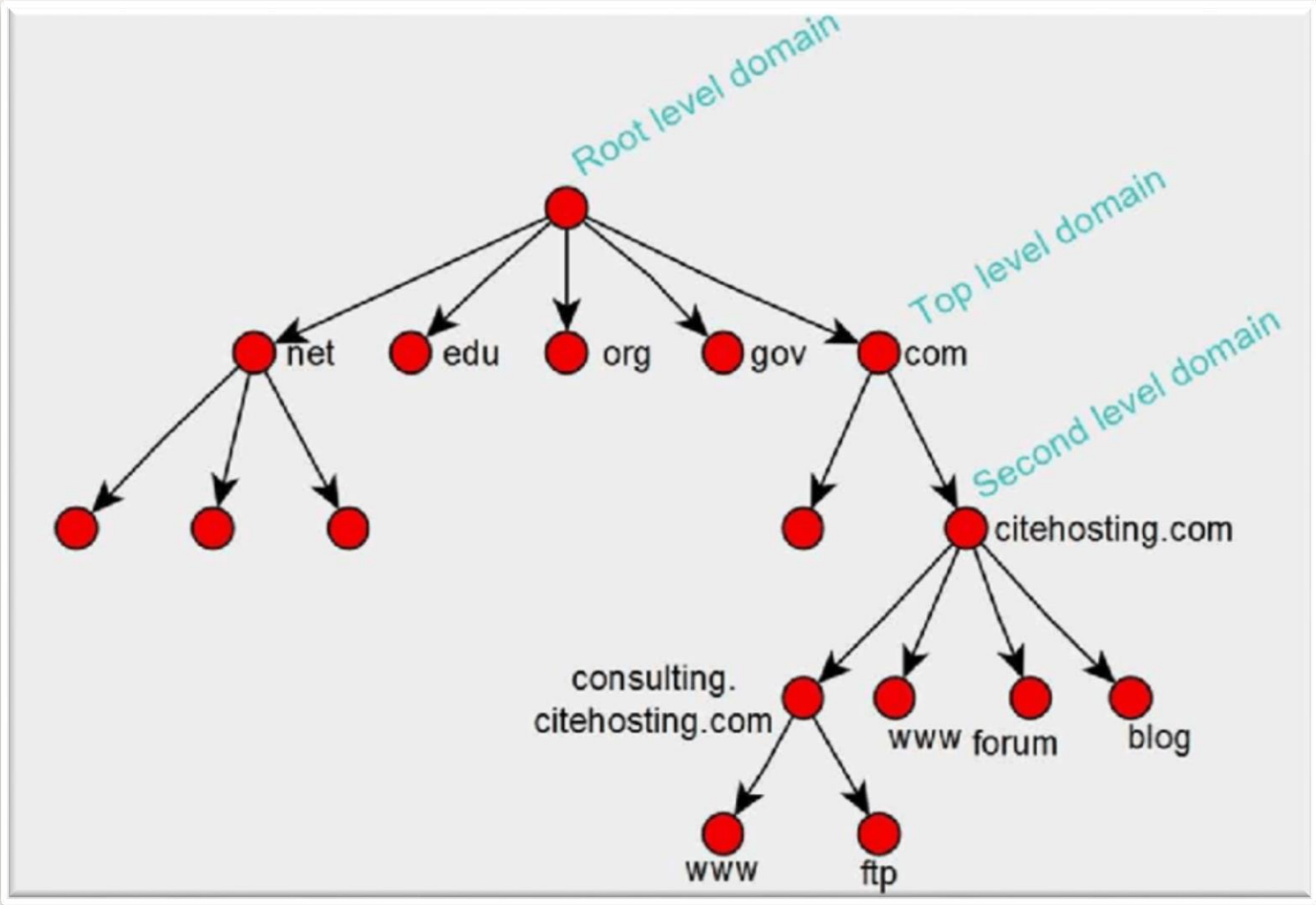
Maps:

- Domain name → IP (forward lookup)
- IP → Domain name (reverse lookup)

Distributed and hierarchical:

- Root servers
- Top-Level Domain (TLD) servers
- Authoritative servers

DNS Hierarchy



Generic domain labels

Label	Description	Label	Description
aero	Airlines and aerospace	**int**	International organizations
biz	Businesses or firms	**mil**	Military groups
com	Commercial organizations	**museum**	Museums
coop	Cooperative organizations	**name**	Personal names (individuals)
edu	Educational institutions	**net**	Network support centers
gov	Government institutions	**org**	Nonprofit organizations
info	Information service providers	**pro**	Professional organizations

DNS Query Process

1. **User types www.example.com**
2. **Local resolver checks cache.**
   - The local resolver (Software that manages the entire lookup process) on the user's computer first checks its **local cache**.
   - **Cache Hit:** If the IP address is in the cache and **hasn't expired** (based on the TTL - Time To Live), it returns the IP immediately. The process stops here. This is the fastest outcome.
   - **Cache Miss:** If it's not in the cache, the resolver must query an external server. It sends a recursive query to the Recursive Resolver (usually your ISP's DNS server or a public one like Google's **8.8.8.8**).

DNS Query Process (Cont.)

3. **The Recursive Resolver's Quest (Root server → TLD → Authoritative name server.)**

   The Recursive Resolver now takes on the job of finding the answer. It is configured to perform **iterative queries** to other DNS servers to fulfill the client's recursive request.

   This is the multi-step process as follows:

   a. The Recursive Resolver first asks a Root Server **(.)**: "Where can I find www.example.com?“ Response: The Root Server doesn't know the answer, but it knows who manages the **.com domain**. It replies with a referral to the **TLD (Top-Level Domain) servers for .com**.

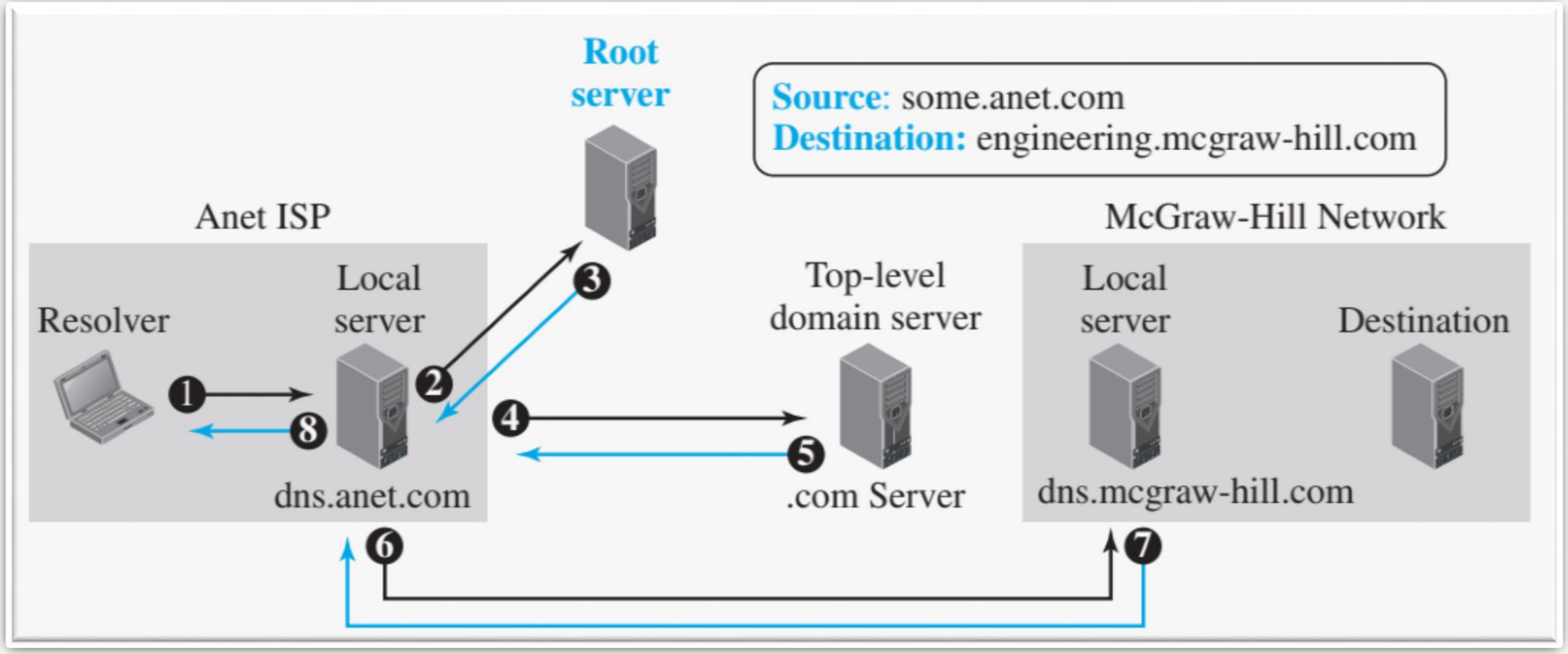
   b. The Recursive Resolver now asks a **.com TLD Server**: "Where can I find www.example.com?“. Response: The TLD Server doesn't know the full answer either, but it knows the **authoritative name servers for the example.com domain**. It replies with a referral to the authoritative name servers for example.com.

DNS Query Process (Cont.)

c. Finally, the Recursive Resolver asks the **Authoritative Name Server for example.com**: "What is the IP address for www.example.com?“ Response: This server is the ultimate source of truth for the example.com domain. It looks up the www record and returns the final IP address (e.g., 93.184.216.34).

4. **IP address returned to client and cached.**

DNS Query Process (Cont.)



DNS Records

Record	Meaning	Example
A	IPv4 address	www.example.com → 192.168.1.10
AAAA	IPv6 address	example.com → 2001:db8::1
CNAME	Alias	www → server1.example.com
MX	Mail exchanger	mail → 192.168.1.20
NS	Name server	example.com → ns1.example.com
PTR	Reverse lookup	192.168.1.10 → www.example.com

DNS Caching & TTL

- **Local Caching:** DNS results stored by your device/router/ISP after first lookup
- **TTL Control:** Time-To-Live value dictates **maximum cache duration** in **seconds**
- **Traffic Reduction:** Prevents repeated queries for same domain, reducing DNS server load
- **Speed Boost:** Cached responses are instant vs. full resolution (100ms+)
- **Propagation Delay:** When IP changes, users see old IP until TTL expires globally
- **TTL Strategy:**
  - **Short TTL** (minutes): **Fast changes, failover, load balancing**
  - **Long TTL** (hours/days): **Stable services, performance**
- **Bottom Line:** **Caching makes DNS fast and efficient, but requires careful TTL management when making changes.**

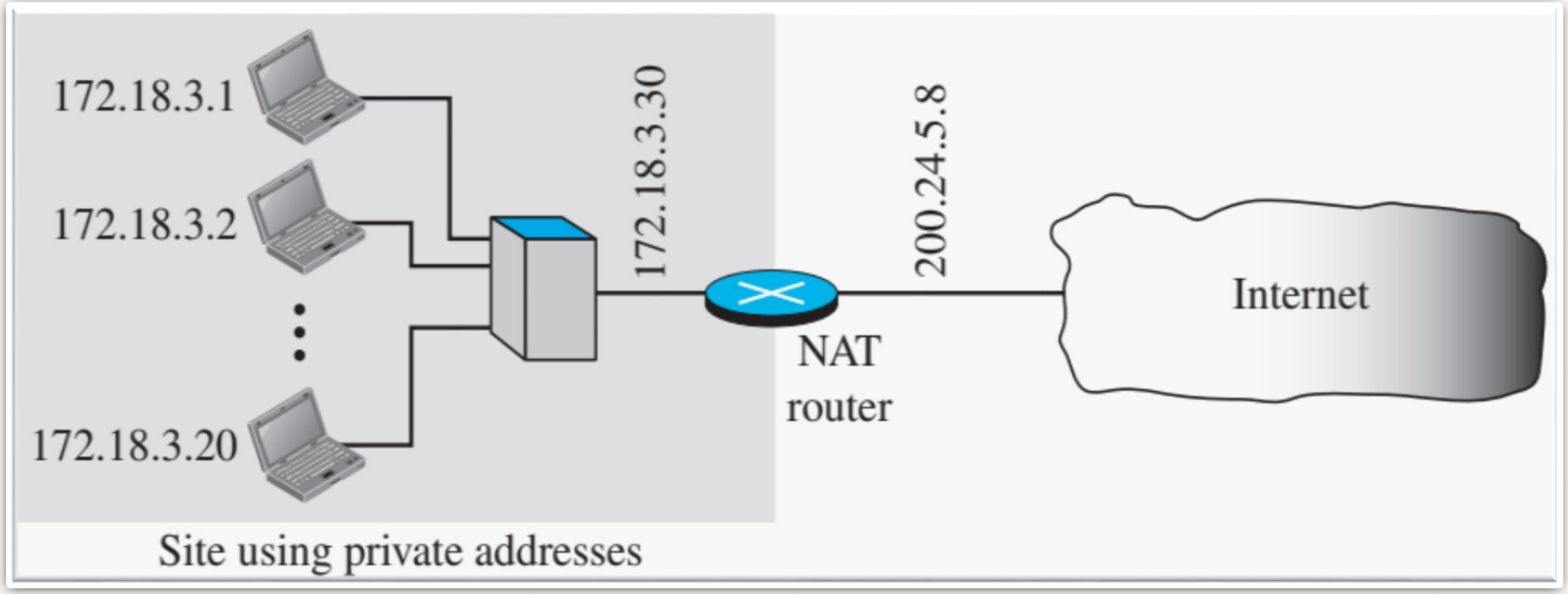
NAT

Network Address Translation (NAT)

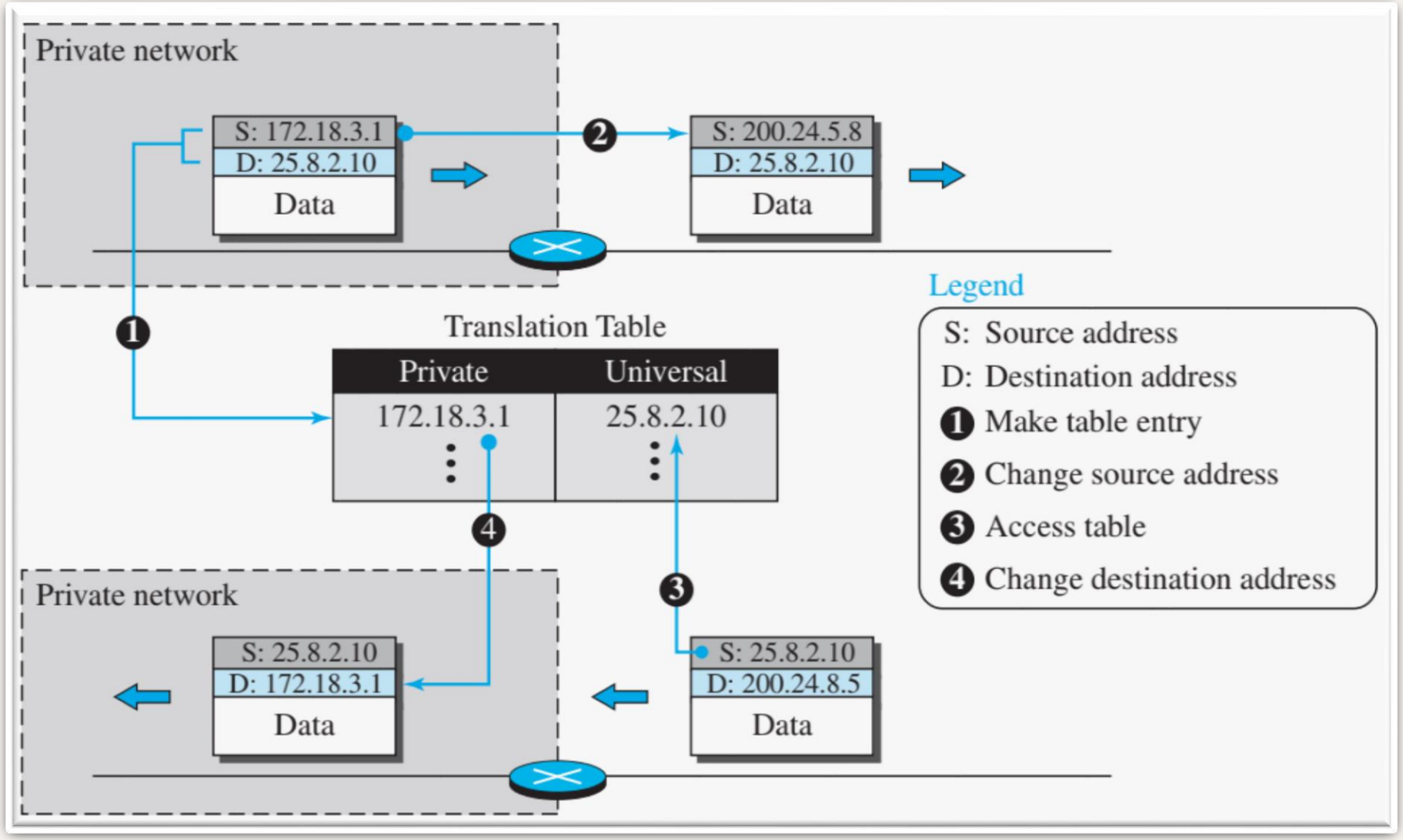
- **IPv4** has **only 4.3 billion unique addresses**, but **billions of devices exist**.
- **Private networks (LANs) reuse RFC 1918 address ranges:**
  - **10.0.0.0/8**
  - **172.16.0.0/12**
  - **192.168.0.0/16**
- **However, private IPs cannot be routed on the Internet.**
- **NAT allows multiple private devices to share one public IP by translating addresses.**

NAT (Cont.)

NAT allows a site to use a set of **private addresses** for internal communication and a set of **global Internet addresses** (at least one) for communication with the rest of the world.



Translation



How NAT Works - Step by Step

Outbound Traffic (Private → Public)

1. Internal Device (192.168.1.10) sends a packet to 93.184.216.34 (example.com)
2. NAT Device intercepts the packet and:
    a. Changes Source IP from 192.168.1.10 to public IP 203.0.113.5
    b. Changes Source Port from 54321 to an available external port like 62001
    c. Creates an entry in the translation table
3. Packet Sent to internet with source: **203.0.113.5 : 62001**

How NAT Works - Step by Step (Cont.)

Inbound Traffic (Public → Private)

1. Response arrives for **203.0.113.5:62001**
2. NAT Device checks translation table:
   - Finds that **203.0.113.5:62001** maps to **192.168.1.10:54321**
3. NAT Rewrites the packet:
   - Changes Destination IP to **192.168.1.10**
   - Changes Destination Port to **54321**
4. Packet Forwarded to the correct internal device

Types of NAT

1. **Static NAT**
   - One-to-one fixed mapping
   - Example: 192.168.1.10 → 203.0.113.10 (always)
   - Useful for hosting servers inside your network

2. **Dynamic NAT**
   - Pool of public IPs used for multiple internal devices
   - Useful for medium-sized organizations

3. **PAT (Port Address Translation) / NAT Overload**
   - The most common type (used in homes)
   - Many private IPs → One public IP
   - Uses port numbers to track connections
   - This is what the translation table example above demonstrates

Benefits of NAT

- **Solves IPv4 Address Exhaustion**
  - Dozens of devices can share one public IP
- **Security Through Obscurity:**
  - Internal network structure is hidden from the internet
- **Network Simplicity:**
  - Easy to change ISPs without renumbering entire internal network

Limitations of NAT

1. **Breaks End-to-End Connectivity**
   - ➢ Makes peer-to-peer applications and hosting servers difficult
2. **Complexity: Some protocols (like FTP)**
   - ➢ Sometimes this protocol embed IP addresses in their data, requiring special NAT helpers
3. **Performance Overhead**
   - ➢ Each packet must be examined and modified
4. **Hides Multiple Users**
   - ➢ Makes user-based tracking and filtering challenging for organizations