DNS Query Resolution - Complete Study Notes

Overview

DNS (Domain Name System) queries can be resolved using two methods:

1. **Recursive Method**
2. **Iterative Method**

DNS Query Resolution Process

Initial Setup

Scenario: Opening a browser and typing a URL

- **Example**: `http://www.cricinfo.com`
- **Domain Name**: cricinfo (needs corresponding IP address)
- **Domain Type**: .com (commercial domain)

Step 1: Local Cache Check (DNS Resolver)

- First, search for the IP address of cricinfo in **local cache** (DNS resolver)
- **DNS Caching**: Prevents requests from going to root servers repeatedly
- **Implementation**: ISP maintains local cache with most popular/frequently used domain mappings
- **Best Case**: If cricinfo's IP is found in cache, query is resolved immediately
- **Cache Miss**: If accessing a site for the first time, DNS won't be in cache - must query root servers

---

Recursive Method

How Recursive DNS Works

Step 1: Search in local cache

- If not found, proceed to proper searching

Step 2: Request goes to Root Server

- Root server identifies the domain type (.com = commercial)
- Directs to appropriate Top Level Domain (TLD) server

Step 3: Request forwarded to TLD Server

- Generic names server for .com domains

Step 4: Request forwarded to Authoritative Server

- TLD server decides which authoritative server to contact
- Authoritative servers contain actual IP addresses

Step 5: IP Address Retrieved

- Authoritative server provides the IP address
- IP address received but hasn't reached the client yet

Step 6: Response back to Root Server

- Authoritative server sends IP back through the chain

Step 7: Root Server Responds

- Root server sends response back to DNS resolver

Step 8: Client Receives IP Address

- Finally receive IP address of cricinfo
- Machine can now locate the target network using network ID and host ID
- Connect to the specific network and retrieve data

Characteristics of Recursive Method

- Root server maintains responsibility throughout the entire process
- Root server must handle the request again when response comes back
- More load on root server

Iterative Method

How Iterative DNS Works

Step 1: Search in local cache

- Same as recursive: if found, query resolved

Step 2: Request to Root Server

- Root server identifies domain type (.com = commercial)
- **Key Difference**: Root server provides TLD server address and its IP
- Root server's job ends here (responsibility reduced)

Step 3: DNS Resolver contacts TLD Server directly

- Using the IP address provided by root server
- DNS resolver makes direct request to TLD server

Step 4: TLD Server Response

- TLD server identifies cricinfo's authoritative server
- Provides authoritative server's address to DNS resolver

Step 5: Direct Request to Authoritative Server

- DNS resolver contacts authoritative server directly
- No intermediate forwarding required

Step 6: Final Response

- Authoritative server provides cricinfo's IP address
- IP address returned directly to client

Characteristics of Iterative Method

- Reduced responsibility on root server
- Root server's work ends after first response
- More direct communication between DNS resolver and servers

Key Differences Between Methods

Aspect	Recursive	Iterative
Root Server Load	Higher - maintains responsibility throughout	Lower - work ends after first response
Communication Pattern	Through intermediate servers	Direct communication
Response Path	Back through the chain	Direct responses
Server Responsibility	Root server handles return requests	Each server responds directly

---

DNS Protocol Choice

Why DNS Uses UDP Protocol

Question: Does DNS use TCP or UDP protocol?

Answer: DNS uses **UDP Protocol**

Reasons for UDP Selection

1. **Speed Requirement**

    - UDP is faster than TCP
    - No connection establishment overhead

2. **TCP Limitations**
    - TCP uses three-way handshaking
    - Adds unnecessary delay for DNS queries

3. **Backend Efficiency**
    - DNS resolution happens in background
    - **Primary Goal**: Fast access to actual content (cricinfo match results)
    - **Secondary Goal**: Quick DNS resolution (should be invisible to user)

4. **User Experience**
    - Main objective: View website content quickly
    - DNS resolution should not create bottlenecks
    - Backend processes must be optimized for speed

Protocol Comparison for DNS

- **TCP**: Slower due to connection establishment, but reliable
- **UDP**: Faster, connectionless, perfect for quick DNS lookups

---

Practical Example Walkthrough

Scenario: First-time visit to cricinfo.com

1. **User Action**: Types `http://www.cricinfo.com` in browser
2. **Cache Check**: No entry found (first visit)
3. **Method Selection**: Either recursive or iterative
4. **DNS Resolution**: Following chosen method steps
5. **IP Retrieved**: e.g., 192.168.1.100 (example IP)
6. **Network Location**: Using network ID and host ID from IP
7. **Data Retrieval**: Connect to cricinfo server and load website

Network Concepts Applied

- **IP Address Structure**: Network ID + Host ID
- **Network Location**: Server could be anywhere globally
- **Data Access**: Navigate to specific network and retrieve website data

---

Summary

DNS query resolution is fundamental to internet browsing, with two main approaches offering different trade-offs between server load and communication efficiency. The choice of UDP protocol ensures fast resolution times, making the DNS lookup process transparent to end users who primarily want quick access to website content.

Both recursive and iterative methods achieve the same goal through different communication patterns, with iterative method being more efficient for root server load distribution in large-scale DNS infrastructure.