University of Dhaka

Department of Computer Science and Engineering

CSE 3111 – Computer Networking Laboratory Credits: 1.5 Batch: 27/3rd Year 1st Sem 2023

Instructors: Prof. Dr. Md. Abdur Razzaque (AR), Prof. Dr. Md. Mamun-Or-Rashid (MOR),
Dr. Muhammad Ibrahim (MIb) and Mr. Md. Redwan Ahmed Rizvee (RAR)

Lab Experiment # 4

Title of the Experiment: Distributed Database Management, Implementation of Iterative, and Recursive Queries of DNS Records.

Objective:

The preliminary objective of this lab is to emulate the Domain Name Service (DNS) protocol and to understand the difference between iterative and recursive DNS resolution. The client can request IP address of his desired domain and the nameserver hierarchy will use the DNS resolution to return the IP address of the corresponding domain to the client if the domain name is valid.

Theory:

When we visit a domain such as google.com, our computer follows a series of steps to convert the human-readable web address into a machine-readable IP address. IP address retrieval process is as follows:

1. Search in DNS cache: When we ask our computer to resolve a hostname, the first place our computer looks is in its local DNS cache. If our computer doesn't already know the answer, it needs to perform a DNS query to find out.

2. Request to ISP's DNS servers: If the information is not stored locally, our computer queries our ISP's DNS servers.

3. Request to root nameservers: If those servers don't have the answer, they query the root nameservers. A nameserver is an always running computer that resolves queries about domain names, such as IP addresses.

4. Request to Top Level Domain nameservers: The root nameservers will look at the first part of our request, reading from right to left and direct our query to the Top-Level Domain (TLD) nameservers for .com (for google.com). Each TLD, such as .com, .org, and .bd, have their own set of nameservers, which act like a receptionist for each TLD.

5. Request to authoritative DNS servers: The TLD nameservers review the next part of our request and direct our query to the nameservers responsible for this specific domain. These authoritative nameservers are responsible for knowing all the information about a specific domain, which are stored in DNS records.

6. Retrieve the record: ISP's DNS server retrieves the record for google.com from the authoritative nameservers and stores the record in its local cache. If anyone else requests the host record for google.com, the ISP's servers will already have the answer and will not need to go through the lookup process again. All records have an expiration time. After a while, the server will need to ask for a new copy of the record to make sure the information doesn't become out-of-date.

7. Receive the answer: Armed with the answer, ISP's server returns the record back to our computer. Computer stores the record in its cache, reads the IP address from the record, then passes this information to browser. The browser then opens a connection to the webserver and receives the website.

Methodology :

1. We create a thread for each DNS server.
2. We create a tree of DNS servers with a root node.
3. Each server is provided with the reference to its parent server and child servers.
4. The server awaits for request from client
5. After receiving request, server looks for the requested domain in its current location and the child (if any). if not found, it will forward the request to the parent.
6. Parent does a similar search, if not found, it sends the request to its parents.
7. If the domain can't be found in any of the DNS servers, then the client will be prompted with an error message.
8. If the domain is found in any of the servers, they will return to the requesting client. (Iteratively/recursively)

Tasks:

Part 1: Setting up the DNS server

1. Configure the DNS server to act as an authoritative server for a domain (e.g., cse.du.ac.bd).
2. Add A, AAAA, CNAME, and MX records for the domain in a file.
    - Use IP address of your friends' PC as the IP address and their name as the domain name.
    - We have added an example file dns_records.txt. This file represents a simple DNS zone file for the domain "cse.du.ac.bd". It includes DNS records for the domain in (Name, Value, Type, TTL) format.
3. Start the DNS server.
4. Verify that the DNS server is running and that it can resolve queries for the domain when requested by a client via **UDP socket** [Hint: `DatagramSocket` and `DatagramPacket` classes provided by the `java.net` package]
5. For exchanging message between DNS server and client, use the following format:
6. Modify the DNS server to perform as root and TLD server.

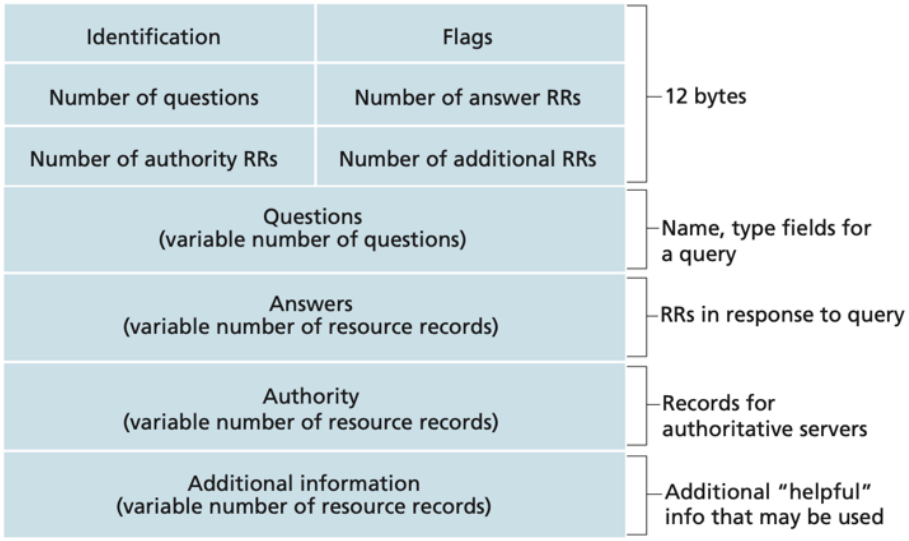

Figure 2.21 ♦ DNS message format

Part 2: Iterative DNS resolution

1. Write a script that sends a DNS query to the root DNS server.
2. The root DNS server will respond with a referral to a top-level domain (TLD) DNS server.
3. The script will then send the query to the TLD DNS server.
4. The TLD DNS server will respond with a referral to the authoritative DNS server for the domain.
5. The script will then send the query to the authoritative DNS server.

6. The authoritative DNS server will respond with the IP address for the domain.
7. Verify that the script correctly implements iterative DNS resolution.

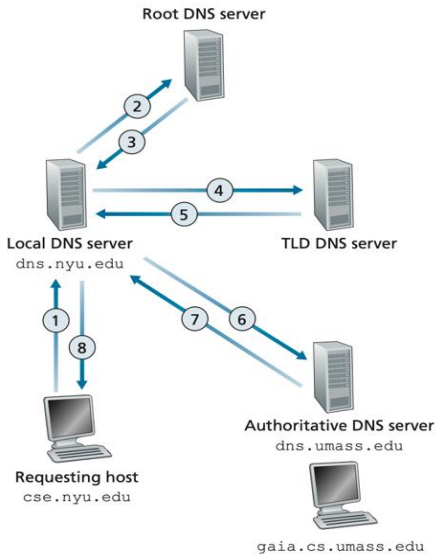

Figure 1.1 : Iterative DNS Queries

Part 3: Recursive DNS resolution

1. Modify the script from Part 2 to send a recursive DNS query to a recursive DNS resolver.
2. The recursive DNS resolver will send queries to the root, TLD, and authoritative DNS servers on behalf of the script.
3. The recursive DNS resolver will respond with the IP address for the domain.
4. Verify that the script correctly implements recursive DNS resolution.

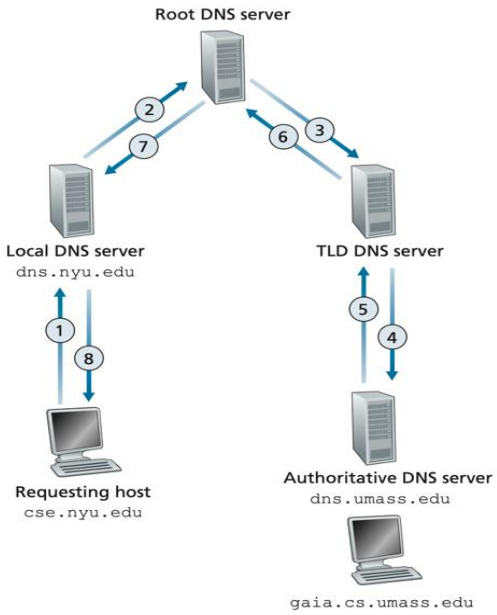

Figure 2.2 : Recursive DNS Queries

Part 4: Extending the System

1. Use a short TTL value and try Deleting resource record based on TTL value.
2. Implement DNS caching in local and TLD servers
3. Test failure of a DNS server process.

Deliverables:

- A report detailing
    1. the steps taken to set up the DNS server and implement the iterative and recursive DNS resolution methods.
    2. Comparison of the time required for resolving a request via iterative and recursive DNS resolution methods. Do this for different scenarios.
    3. advantages and disadvantages of each method and the security implications of each method.
- Source code zip for each of the 4 tasks.

Reference :

1. DNS Basic - https://aws.amazon.com/route53/what-is-dns
2. Record Types - https://constellix.com/news/dns-record-types