DNS Resolver

Kumar Kanishk Singh (210544)
Sunny Raja Prasad (218171078)
Tanmey Agarwal (211098)

March 2, 2025

1 Overview

This script implements both iterative and recursive DNS resolution methods to resolve domain names to IP addresses. The script queries root DNS servers and follows the DNS resolution hierarchy to find authoritative name servers that provide the final response.

2 How to Run

The script can be executed in two modes:

- Iterative DNS resolution
- Recursive DNS resolution

Use the following command to run the script:

```
python3 dns_server.py <mode> <domain>
```

mode: Choose `iterative` for iterative DNS resolution or `recursive` for recursive resolution.
domain: The domain name you want to resolve (e.g., example.com).

2.1 Example Usage

Iterative DNS Resolution:

```
python3 dns_server.py iterative example.com
```

Recursive DNS Resolution:

```
python3 dns_server.py recursive example.com
```

3 How It Works

3.1 Iterative Mode

- Choose any root server from the available ones, send a UDP query to it, and retrieve various NS hostnames.
- Use `dns.resolver.resolve` to find the corresponding IP addresses of the TLD servers and update the `nxt nx list` array accordingly.
- Repeat the same process by selecting a TLD server, sending a UDP query to obtain the IP addresses of the authoritative name servers, and updating the `nxt ns list` array accordingly.
- After obtaining the final set of servers to query, send a query to retrieve the final IP address.
- During intermediate steps, handle and check for any exceptions.

3.2 Recursive Mode

- Use the `dns.resolver.resolve` method from the dnspython library to obtain NS record hostnames.
- Retrieves NS and A records for the given domain.
- Prints the resolved IP address or an error message if resolution fails.

4 Error Handling

- Handles timeouts, bad responses, transfer errors, and other unexpected failures.
- Prints appropriate error messages to help debug resolution issues.
- Uses a timeout of 7 seconds per query attempt to prevent excessive delays.

4.1 What more could be done

- Currently, the program terminates if no valid response is received at any stage. Ideally, it should attempt querying the next available servers if a bad response is encountered.
- Cache DNS responses to reduce delay time for repeated DNS queries.
- Perform parallel queries by utilizing multiple servers to resolve domain names simultaneously.

5 Individual Contributions

5.1 Kumar Kanishk Singh (210544)

- Implemented iterative DNS resolution
- Debugging and testing

5.2 Sunny Raja Prasad (218171078)

- Implemented recursive DNS resolution
- Wrote and formatted the README documentation

5.3 Tanmey Agarwal (211098)

- Handled exception cases and error handling
- Assisted in debugging and refining code

6 Sources

The following sources were referred while solving the assignment:

- dnspython documentation
- IANA Root Servers Information
- Various online blogs and tutorials on DNS resolution

7 Declaration

We declare that we have not indulged in plagiarism while completing this assignment. All work has been done independently with references cited appropriately.