A blue and white logo

Description automatically generated

CSE 351 Computer Networks  
Fall 2024/2025

Submitted to:

Dr. Ayman Bahaa

Eng. Noha Wahdan

Submitted by:

Mohamed Mostafa Mamdouh 21P0244

Ali Tarek 21P0123

Abdelrahman Sherif Hassan

Gaser Zaghloul

Table of Contents

[Introduction 3](#_Toc183432129)

[Required RFCs 3](#_Toc183432130)

[RFC 1034 3](#_Toc183432131)

[RFC 1035 3](#_Toc183432132)

[RFC 2181 3](#_Toc183432133)

[Project Planning and Design 4](#_Toc183432134)

[System Architecture 4](#_Toc183432135)

[Communication Protocols 5](#_Toc183432136)

[Steps 5](#_Toc183432137)

# 

# Introduction

The goal of this project is to analyze and implement a server agent for DNS protocol that is compliant with the RFC. The project should work seamlessly with common off-the shelf clients such as a typical browser or Windows DNS client agents. The project will be divided into four phases, each building on the previous one, to demonstrate compliance details with the RFC. The project should follow behaviors, scenarios, messages format and sequence, and produce RFC-compliant outcomes or error codes.

Required RFCs

RFC 1034 Domain Names – Concepts and Facilities. This foundational RFC defines the concept of the Domain Name System (DNS), which is responsible for translating human-readable domain names into IP addresses. It outlines the basic structure and operation of DNS.

* Link: [RFC 1034 - Domain names - concepts and facilities](https://datatracker.ietf.org/doc/rfc1034/)

RFC 1035 Domain Names – Implementation and Specification. This RFC provides detailed specifications for the DNS protocol, including message formats, query types, and response structures. It focuses on how the DNS protocol operates in practice.

* Link: [RFC 1035 - Domain names - implementation and specification](https://datatracker.ietf.org/doc/rfc1035/)

RFC 2181 Clarifications to the DNS Specification. This RFC clarifies ambiguities and updates certain aspects of the original DNS specification (RFC 1034 and RFC 1035) to address issues encountered during deployment.

* Link: [RFC 2181 - Clarifications to the DNS Specification](https://datatracker.ietf.org/doc/rfc2181/)

# Project Planning and Design

The project scope involves developing a DNS server agent that works seamlessly with common DNS clients like Windows DNS client or dig.

Its goal involves ensuring compliance with DNS protocol RFCs 1034, 1035 and 2181, and be able to handle valid queries and errors effectively.

The DNS server is the core component of the architecture that processes DNS queries and generates responses. It includes modules for parsing queries, accessing stored data, handling errors and constructing RFC-compliant responses.

This server will perform the following key functionalities:

* Resolve domain names to IP addresses (forward lookup).
* Translate IP addresses back to domain names (reverse lookup).
* Provide error codes and responses as per the DNS specifications.
* Support typical DNS query types (A, AAAA, CNAME, MX, etc.).
* Handle query and response messages in a robust, RFC-compliant manner.

## System Architecture

DNS server agent will consist of the following components:

1. **Query Processor**: Parses incoming DNS queries and determines their type (e.g., A, AAAA, MX, CNAME). It validates the structure of the query based on RFC 1035. It extracts the domain name from the question section, identifies the query type and class then validates packet headers for correctness.

**Example**: For a query requesting the IPv4 address of example.com, the Query Processor extracts the domain and type (A record).

1. **Database**: Stores domain-to-IP mappings and related DNS records, such as TTL (Time to Live), record types, and reverse mappings (PTR).
   1. Forward Lookup: Resolves domain names (e.g., example.com) to IPs (e.g., 93.184.216.34).
   2. Reverse Lookup: Resolves IPs to domain names using PTR records.
2. **Response Generator**: Constructs responses to queries, adhering to the DNS message format specified in RFC 1035.
   * Adds the Answer Section with the resolved data if a match is found in the database.
   * Fills other sections (Authority and Additional) if needed.
   * Handles compression for efficient response transmission.
3. **Error Handler**: Handles errors by generating appropriate DNS response codes (RCODEs).

* **Details**:
  + NXDOMAIN (3): No such domain.
  + SERVFAIL (2): General failure (e.g., database issue).
  + REFUSED (5): Server refuses to answer (e.g., unauthorized).

## Communication Protocols

Both queries and responses follow the structure defined in RFC 1035, including the following sections:

* **Header**: Contains metadata like transaction ID, flags, and query count.
* **Question**: Specifies the domain name and query type (e.g., A, AAAA, MX).
* **Answer**: Contains the resolved data (e.g., IP address).
* **Authority**: Points to authoritative name servers.
* **Additional**: Provides extra information (e.g., IP of name servers).

Responses include a TTL field, which specifies how long clients can cache the data.

**UDP** will be used as the default transport protocol on port 53, which is used for most DNS queries due to its low latency.

**Workflow Example**

1. Client sends a query: "What is the IP address of example.com?"
2. Query Processor parses the query and checks the database.
3. Database returns the IP address (93.184.216.34).
4. Response Generator constructs a response and includes the Answer Section with the IP.
5. Error Handler intervenes if the query is invalid or the domain is not found.
6. The response is sent back to the client using UDP.

## Steps

1. **Define User Authentication Mechanisms**

Objective here is to restrict access to sensitive operations like zone transfers or dynamic DNS updates. This can be done through the use of **ACLs** (Access Control Lists), where rules are placed on client IP address to allow or deny specific actions.   
Only IPs in a trusted subnet can perform updates.  
Public clients can query but not modify DNS records.

1. Outline the overall