HTTP-Proxy

Amzuloiu Sergiu
Racianu Marian Gabriel
Grupa C113E

January 17, 2024

1 Table of Contents

Contents

2 Version Table

Date	Version	Notes
7/11/2023	1.0	- Created repository - Added first version of product requirements document
20/11/2023	1.1	- Managed a single connection client-server
1/12/2023	1.2	- Managed multiple connections client-server
22/12/2023	2.0	- Received first web server response
28/12/2023	2.1	- Added parsing functionality
7/01/2024	2.2	- Implemented caching
15/01/2024	2.3	- First version of frontend

Table 1: Version History

3 Introduction

3.1 Project Overview

proxydock is a software application designed to act as an intermediary between clients and servers over the HTTP protocol. It facilitates the forwarding of HTTP requests and responses, enabling users to access web content securely and anonymously. This software application acts as a reliable gateway; proxydock ensures communication between clients and servers while maintaining the utmost privacy and confidentiality for the end-users, also providing a GUI for easier use.

3.2 Objective

The fundamental objective of proxydock is to act as a stand-in user, effectively preserving the privacy, confidentiality of the real client and allowing him to track the traffic. The application allows users to browse the internet, access web services, and communicate with servers, while providing useful functionalities.

4 Functional Requirements

4.1 Proxy Server Configuration

- Users should be able to configure proxy server settings, including hostname, port, and protocol.

4.2 Request Routing

- The app should route HTTP requests from clients to the specified destination server.

4.3 Caching

- Ability to cache responses to reduce latency and improve performance.

4.4 Compression and Decompression

- Support for compressing and decompressing HTTP content, such as GZIP.

4.5 Protocol Support

- Handle various HTTP and proxy protocols, including HTTP/1.0, HTTP/1.1, and HTTP/2.

4.6 Content Filtering

- Offer content filtering options to block or filter specific websites or content types.

4.7 Traffic Inspection and Manipulation

- Allow for the inspection and manipulation of HTTP requests and responses, which can be useful for debugging or security purposes.
- Editing/adding headers.

5 Non-Functional Requirements

5.1 Performance

5.1.1 Response Time

- Define the maximum acceptable response time for processing requests.

5.1.2 Throughput

- Specify the desired level of concurrent connections and requests the proxy should handle.

5.1.3 Scalability

- Determine the ability of the app to scale with increased demand.

5.2 Interoperability

- Ensure that the app can work with a variety of web browsers, operating systems, and other HTTP clients.
- Support for industry standards and protocols.

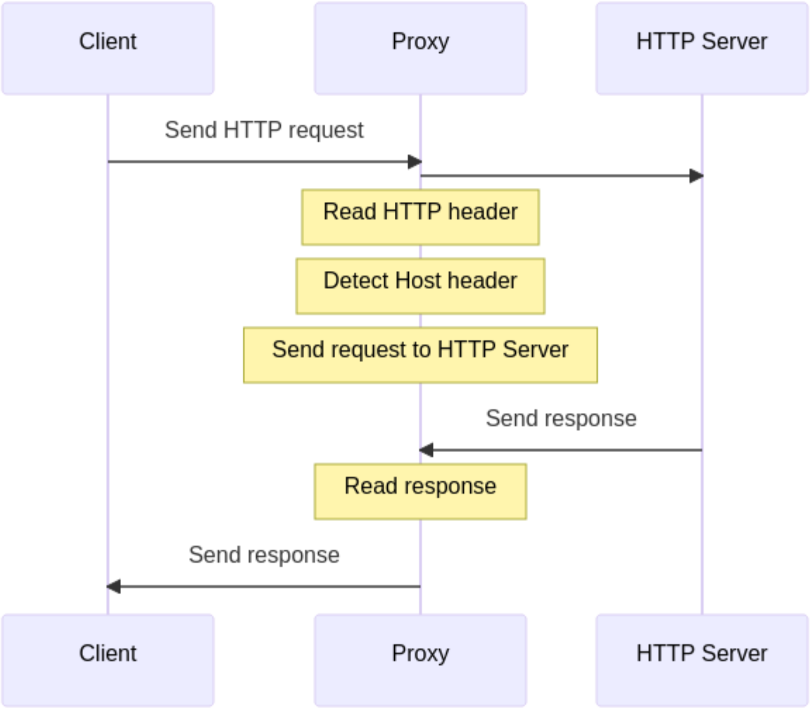
5.3 Documentation

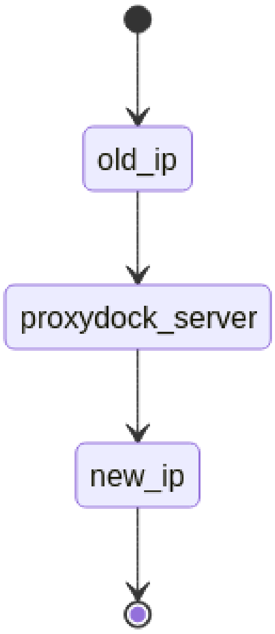
- Specify documentation requirements, including user manuals, API documentation, and code comments.

5.4 Compliance with Web Standards

- Ensure that the app complies with HTTP and web standards, including HTTP/1.1, HTTP/2, and WebSocket protocols.

6 Architecture





7 Technologies used

C/C++/Qt

Project completed within the PSO course by Amzuloiu Sergiu and Racianu Gabriel under the guidance of Vaman Adina.