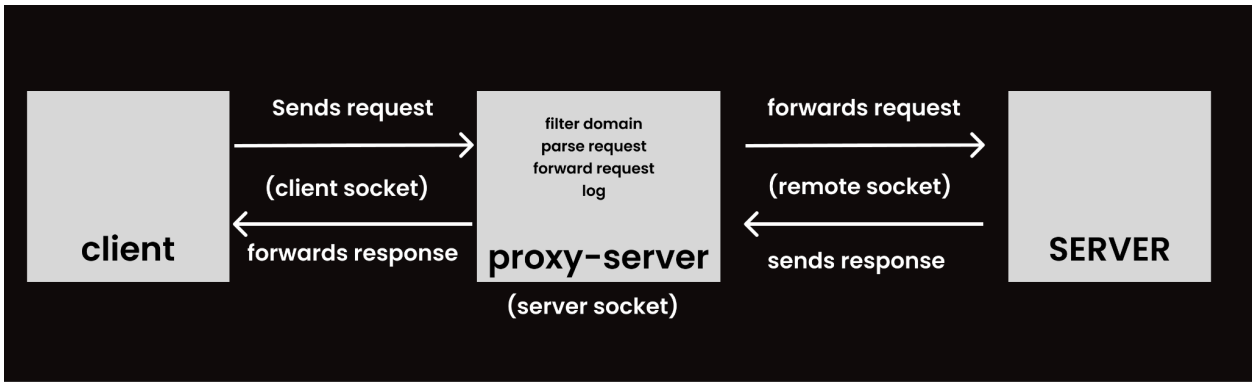
High-Level Architecture



The system consists of three main components: Client, Proxy Server, and Destination Server. Clients send HTTP requests to the proxy, which enforces filtering policies and forwards requests to destination servers.

Concurrency Model

The thread-per-connection model was chosen due to its simplicity and ease of implementation. It allows independent handling of each client request and is suitable for moderate concurrency levels, which aligns with the educational and prototype nature of this project.

Request Handling and Data Flow

The proxy reads incoming HTTP requests, parses the destination host and path, applies filtering rules, forwards allowed requests, streams responses back to clients, and logs request metadata.

HTTP Request Parsing

The proxy supports both absolute URI requests and relative path requests with Host headers. It extracts method, host, port, and path for correct forwarding.

Forwarding Mechanism

For allowed requests, the proxy establishes a TCP connection to the destination server, rewrites the request into origin-server format, and streams the response back to the client without full buffering.

Filtering and Configuration

Filtering rules are stored externally in a configuration file containing blocked domains or IPs. Blocked requests receive a 403 Forbidden response.

Logging

Each request is logged with timestamp, client address, destination host, request line, action taken, HTTP status code, and number of bytes transferred.

Error Handling

The proxy handles partial TCP reads, malformed requests, and ensures proper socket cleanup to avoid resource leaks.

Limitations

The proxy supports only HTTP traffic and does not implement HTTPS CONNECT tunneling, caching, or authentication. Chunked transfer encoding is forwarded transparently without decoding.

Security Considerations

The proxy does not inspect encrypted payloads, does not store response content, and limits logging to metadata only.

Future Extensions

Possible extensions include HTTPS CONNECT support, caching, wildcard domain filtering, and improved concurrency models.

Conclusion

The project satisfies all core objectives of the specification and demonstrates key networking and systems concepts through a functional and extensible proxy server implementation.Design and Implementation of a Custom Network Proxy Server 1. Overview This project implements a forward HTTP proxy server that acts as an intermediary between clients and destination servers. The proxy accepts TCP connections, parses HTTP requests, applies access-control rules, forwards allowed traffic, and relays responses back to clients. 2. High-Level Architecture The system consists of three main components: Client, Proxy Server, and Destination Server.