BÁO CÁO BÀI THỰC HÀNH – XÂY DỰNG MÁY CHỦ HTTP BẰNG C

1. Giới thiệu

Bài thực hành yêu cầu xây dựng một HTTP server đơn giản bằng ngôn ngữ C, có khả năng xử lý nhiều khách hàng đồng thời thông qua fork(), phục vụ các tệp HTML và CSS, đồng thời xử lý các lỗi 404 (Not Found) và 405 (Method Not Allowed).

2. Mô tả triển khai

- Khởi tạo socket: tạo socket IPv4, gán địa chỉ bất kỳ trên cổng 8080, rồi listen().

- Xử lý đa tiến trình: với mỗi accept(), gọi fork(). Con sẽ đảm nhận serve\_client(), cha đóng socket con và tiếp tục accept.

- Phân tích yêu cầu: đọc toàn bộ header, dùng sscanf() để lấy METHOD và URI từ dòng đầu.

- Chỉ hỗ trợ GET: các method khác trả về 405.

- Xác định đường dẫn file: "/" => ./index.html; các URI khác ghép vào thư mục gốc.

- MIME type: xác định tuỳ theo phần mở rộng .html hoặc .css.

- Phục vụ file: mở file, dùng stat() lấy kích thước, gửi header và đọc – gửi nội dung theo block.

- Xử lý lỗi:

- File không tồn tại: trả về 404.

- Method khác GET: trả về 405.

- Dọn dẹp: con đóng socket, exit; cha dùng signal(SIGCHLD) để tránh zombie.

3. Biên dịch và chạy

gcc server.c -o server

./server

Khi chạy, sẽ in:

Server running on port 8080

4. Kiểm tra máy chủ

- curl http://localhost:8080/ -> trả về nội dung HTML

- curl http://localhost:8080/assets/style.css -> trả về CSS

- curl -i http://localhost:8080/unknown.html -> trả về 404

- curl -X POST -i http://localhost:8080/ -> trả về 405

5. Trả lời câu hỏi (Mục 7)

1) Khi một client kết nối, máy chủ tạo tiến trình con để xử lý yêu cầu đó.

2) Máy chủ xác định loại tập tin qua phần mở rộng (.html, .css).

3) fork() giúp xử lý nhiều kết nối đồng thời mà không làm block.

4) Nếu tập tin không tồn tại, trả về 404 Not Found.

5) Zombie là tiến trình con đã kết thúc nhưng chưa được thu dọn; dùng signal(SIGCHLD) để tránh.

6. Kết luận

Máy chủ hoạt động ổn định, phục vụ đúng nội dung HTML, CSS và xử lý lỗi cơ bản. Việc sử dụng fork() giúp tăng tính đa nhiệm cho ứng dụng mạng viết bằng C.