Lab3-2 분석

1. lab2-3과 lab3-1 차이 분석

소켓에 사용되는 프로토콜

멀티캐스트 채팅 프로그램: UDP

멀티스레드 기반 채팅 프로그램: TCP

서버-클라이언트 구조 여부

멀티캐스트 채팅 프로그램: 서버-클라이언트 모델 없이도 작동할 수 있습니다

멀티스레드 기반 채팅 프로그램:

서버-클라이언트 모델을 따릅니다. 중앙 서버는 들어오는 연결을 처리하고 각 클라이언트에 대해 별도의 스레드를 유지하여 메시지 전송 및 수신 프로세스를 관리합니다.

구현(다중 프로세스 대 다중 스레드)

멀티캐스트 채팅 프로그램: 여러 접근을 프로세스 별로 나누어 관리하는 다중프로세스 방

멀티스레드 기반 채팅 프로그램: 동시적 접근을 위한 멀티스레딩 방식

lab3-1 코드 분석

1. 주요 기능(`main`):

- **목적:** 서버를 초기화 및 관리하고 들어오는 클라이언트 연결을 처리합니다.
- **주요 작업:**
    - 서버 소켓(`serv_sock`)을 초기화합니다.

- 서버 주소 구조(`serv_adr`)를 설정합니다.
- 서버 소켓을 IP 주소 및 포트에 바인딩합니다.
- 들어오는 클라이언트 연결을 수신합니다.
- 잠재적으로 루프에 있는 클라이언트(`clnt_sock`)로부터의 연결을 허용합니다.
- 연결된 각 클라이언트에 대한 새 스레드를 생성하여 통신(`pthread_create`)을 처리하고 클라이언트 소켓을 스레드 처리 함수에 전달합니다.

클라이언트 핸들러 함수(`handle_clnt`):

- **목적:** 전용 스레드에서 개별 클라이언트와의 상호 작용을 관리합니다.
- **주요 작업:**
  - 클라이언트로부터 메시지를 받습니다.
  - 수신된 메시지를 다른 클라이언트에 배포합니다(`send_msg`).
  - 연결이 끊어지면 클라이언트의 소켓을 닫습니다.
  - 동시성 문제를 방지하기 위해 뮤텍스 사용하여 공유 리소스(예: 클라이언트 소켓 목록)를 관리합니다.

메시지 전송 기능(`send_msg`):

- **목적:** 한 클라이언트에서 받은 메시지를 연결된 다른 모든 클라이언트에게 보냅니다.
- **주요 작업:**
  - 클라이언트 소켓 목록을 반복합니다.
  - 원래 메시지를 보낸 사람을 제외한 각 클라이언트에게 메시지를 보냅니다.
  - 동시 수정으로부터 클라이언트 목록을 보호하기 위해 뮤텍스 잠금(`pthread_mutex_lock` 및 `pthread_mutex_unlock`)을 사용합니다.

4. 오류 처리 함수(`error_handling`):

- **목적:** 서버 운영 중 발생할 수 있는 오류를 처리하는 메커니즘을 제공합니다.
- **주요 작업:**
  - 오류 메시지를 인쇄합니다.
  - 심각한 오류가 발생하면 프로그램을 종료하여 심각한 문제로 인해 서버가 정상적으로 중지되도록 합니다.