Nguyễn Khánh Nam - 20225749

1) UDP

1. Trình bày ý nghĩa và chi tiết Flags trong lệnh recvfrom(), sendto()

recvfrom()

* MSG\_PEEK: Xem trước thông tin nhận được trong Queue của Socket nhưng không lấy / xóa ra khỏi Queue
* MSG\_OOB: Hay dùng cho TCP, yêu cầu gửi dữ liệu khẩn cấp
* MSG\_WAITALL: Block lại function cho tới khi đủ Data được trả về

sendto()

* MSG\_OOB: Gửi dữ liệu khẩn cấp, hay dùng cho TCP, trong UDP không có khái niệm MSG\_OOB
* MSG\_DONTROUTE: Không định tuyến qua cổng trung gian Gateway và chỉ dùng trong LAN không qua Router, gửi data trực tiếp đến các Host trong LAN

1. EMSGSIZE return trong sendto()

Lỗi tin gửi quá lớn để có thể gửi trong cùng lúc 1 Datagram của UDP

1. TH nào nên dừng UDP và tại sao

Cho Streaming Data liên tục và giao thức với DNS.

Vì những trường hợp trên yêu cầu tốc độ cao không cần thiết lập lại đường truyền nếu trong trường hợp bị mất kết nối, người dùng có thể bị đơ một lúc nhưng sẽ nhận được gói tin tiếp theo ngay lập tức khi có lại connection.

1. Miêu tả quy trình hoạt động của UDP client và UDP server trong việc sử dụng UDP Socket

Sau khi cả 2 Socket Client và Server thiết lập giá trị: Server chạy trước Client chạy sau

UDP Server chạy blocking function recvfrom() trước đợi gói tin gửi đến từ UDP Client

UDP Client gửi gói tin tới Server qua blocking function sendto() và bị block cho tới khi nhận lại gói tin từ Server

UDP Server nhận được gói tin và trả về reply cho Client thông qua function sendto() sau khi nhận được giá trị địa chỉ, port Client gửi tới từ trên

UDP Client nhận gói tin từ Server gửi đến qua recvfrom()

2) TCP

1. Trình bày sự khác nhau giữa recvfrom() và recv(), sendto() và send()

recvfrom() và recv()

recvfrom(): Cần tham số địa chỉ vì không có thiết lập kết nối trước đó (trong UDP)

recv(): Không cần tham số địa chỉ vì đã được thiết lập đường truyền kết nối từ trước (trong TCP)

sendto() và send(): Tương tự như trên vì đã có đường truyền kết nối từ trước nên chỉ cần truyền tham số Socket.

1. Trình bày ý nghĩa chi tiết Flags trong recv() và send()

recv()

* MSG\_PEEK: Xem trước thông tin nhận được trong Queue của Socket nhưng không lấy / xóa ra khỏi Queue
* MSG\_OOB: Đọc mọi gói tin khẩn cấp từ Client nếu send() có flag MSG\_OOB
* MSG\_WAITALL: Block lại function cho tới khi đủ Data được trả về

send()

* MSG\_OOB: Gửi dữ liệu khẩn cấp
* MSG\_DONTROUTE: Không định tuyến qua cổng trung gian Gateway và chỉ dùng trong LAN không qua Router, gửi data trực tiếp đến các Host trong LAN
* MSG\_DONTWAIT: return lỗi EAGAIN trong TH send() bị nghẽn traffic đường truyền giữa Client và Server
* MSG\_NOSIGNAL: return lỗi EPIPE trong TH send() đến Server mà Server đã ngắt kết nối từ trước đó hoặc không chạy hàm recv() ở Server.

1. Miêu tả rõ slide 18

Client gửi yêu cầu thiết lập đường truyền từ funct connect() tới Server

Server chạy funct listen() lắng nghe yêu cầu đường truyền từ Client và nếu nhận được yêu cầu gửi trả lại Client chấp nhận

Client return giá trị từ funct connect() và được đưa vào Client Queue trong Server, Server muốn giao tiếp gói tin với Client thì sẽ được gọi lần lượt ra khỏi Queue qua funct accept() ở Server.

1. TH nào nên dùng TCP và tại sao

Gửi Mail, giao tiếp dữ liệu Web với User, truyền hình ảnh.

Vì những TH trên yêu cầu một đường truyền dữ liệu thông suốt và tin cậy có xác nhận qua lại giữa Server và Client khi truyền dữ liệu với nhau và cần kiểm soát lưu lượng dữ liệu truyền.

1. Miêu tả quy trình hoạt động của TCP client và TCP server trong việc sử dụng UDP Socket

Sau khi thiết lập giá trị Socket TCP của Client và Server:

Đặc thù TCP: Client không cần bind() vì giá trị địa chỉ port được Kernel tự động gửi đến Server trong quá trình thiết lập đường truyền

Server bật chế độ lắng nghe yêu cầu thiết lập đường truyền từ Client qua funct listen()

Client gửi yêu cầu đường truyền tới Server qua connect()

Server chấp nhận thiết lập đường truyền với Client qua accept() lấy Client ra từ trong Queue (Trong TH có nhiều Client trong Queue thì cũng gọi lần lượt ra từ funct accept())

Client bắt đầu truyền gửi dữ liệu đến Server qua send() mà không cần cài đặt tham số địa chỉ vì đã có đường truyền thiết lập từ trước

Server nhận dữ liệu qua recv() rồi gửi dữ liệu lại Client qua send() và quay về trạng thái ở funct recv() đợi dữ liệu được truyền tới tiếp theo trên đường truyền

Client nhận dữ liệu từ Server qua recv()

Kết thúc: Đóng Socket của TCP Client và Server qua hàm close() để ngắt kết nối và đóng đường truyền dữ liệu đã thiết lập.